

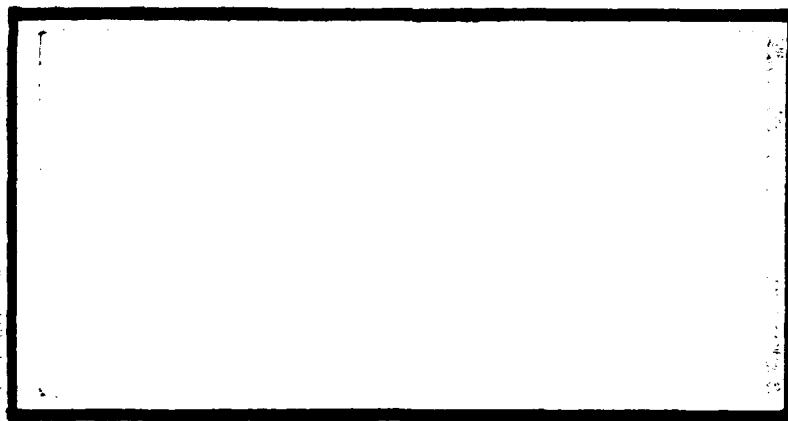
AD-A238 883



1



DTIC
S FILECTE D
JUL 22 1991
D



DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

91-05729



91 7 19 129

①

AFIT/GNE/ENP/91M-10

DTIC
SELECTED
JUL 22 1991
S D D

DEVELOPMENT OF AN ANGULAR DISTRIBUTION FUNCTION
FOR THE STUDY OF ATOMIC LATTICE STRUCTURES
USED IN ATOMISTIC SIMULATION

THESIS

David Wesley, Captain, USAF

AFIT/GNE/ENP/91M-10

Approved for public release; distribution unlimited

91-05729



91 7 10 129

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE March 1991	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE Development of an Angular Distribution Function for the Study of Atomic Lattice Structures Used in Atomistic Simulation			5. FUNDING NUMBERS	
6. AUTHOR(S) David Wesley, Capt, USAF				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583			8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GNE/ENP/91M-10	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) An angular distribution function (ADF) was developed as a tool to be used in atomistic simulation. The ADF lets us easily distinguish between different lattice structures for systems of atoms by looking at the distribution of bond angles. The ADF is also a useful tool for determining the amount of order in a system. First, model structures of simple cubic, FCC, BCC, and diamond were developed. Then, the ADF was used on these model structures so that a set of model graphs of the ADF could be used to compare to real systems. Finally, simulated systems of FeTi, NiTi, and Silicon were analyzed with the ADF. The known structures of the simulated systems all matched their model structures. The ADF was also able to distinguish between liquid and amorphous FeTi, but could not distinguish between liquid and amorphous Silicon. Previously accomplished experiments to induce the transition from crystalline to amorphous with FeTi and NiTi (frenkel pairs and atom exchanges were introduced), were explored using the ADF and it was found that the ADF could be used to interpret the results accurately without the need to compute the structure factor.				
14. SUBJECT TERMS Lattice Structures, Atomistic Simulation, Molecular Bond Angles			15. NUMBER OF PAGES 58	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

AFIT/GNE/ENP/91M-10

DEVELOPEMENT OF AN ANGULAR DISTRIBUTION FUNCTION FOR THE STUDY OF
LATTICE STRUCTURES USED IN ATOMISTIC SIMULATION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology

Air University

In partial Fulfillment of the
Requirements for the Degree of
Master of Science in Nuclear Physics

David Wesley, B.N.E.

Captain, USAF

NAME OF AUTHOR	
DAVID WESLEY	
TITLE	
DEVELOPEMENT OF AN ANGULAR DISTRIBUTION FUNCTION FOR THE STUDY OF LATTICE STRUCTURES USED IN ATOMISTIC SIMULATION	
BY	
DAVID WESLEY	
APPROVED BY	
DAVID WESLEY	
DATE	APPROVED BY
A-1	DAVID WESLEY

March 1991

Approved for public release; distribution unlimited

Preface

The purpose of this thesis was to develop a Fortran program which would analyze the distribution of bond angles between a system of atoms used in atomistic simulation experiments. By analyzing the bond angles it becomes possible to distinguish between lattice structures as well as determine the amount of order in the system. This new angular distribution function should prove to be a useful tool for the study of the crystalline-to-amorphous transition in materials and may also be useful in differentiating between amorphous and liquid systems.

In accomplishing this research I relied a great deal on the expertise and knowledge of my thesis advisor, Capt Michael J. Sabochick. I am indebted to his patience and assistance. I would also like to offer a word of thanks to the Galaxy main-frame computer operators who's success at keeping the computer on-line was immensely helpful. Finally, I wish to thank my wife Pamela for her perseverance under times of great stress. It's not easy to start a marriage when the cards and the school books are stacked against you, but Pamela has been steadfast and marvelous. Thank you Pamela, for the personal sacrifice that you have endured for my sake and for the sake of our marriage.

David Wesley

Table of Contents

1	<u>Introduction</u>	1
1.1	<u>Background</u>	1
1.2	<u>General Approach</u>	2
1.3	<u>Sequence of Presentation</u>	2
2	<u>Structure and Order</u>	3
2.1	<u>Introduction</u>	3
2.2	<u>RDF</u>	3
2.3	<u>Structure Factor</u>	4
2.4	<u>ADF</u>	5
3	<u>Model Systems</u>	12
3.1	<u>Introduction</u>	12
3.2	<u>Results</u>	12
4	<u>Simulated Systems</u>	18
4.1	<u>Introduction</u>	18
4.2	<u>Validation</u>	18
4.2.1	<u>Results</u>	18
4.2.2	<u>Discussion</u>	19
4.3	<u>Liquid versus. Amorphous</u>	21
4.3.1	<u>Results</u>	21
4.3.2	<u>Discussion</u>	21
4.4	<u>Frenkel Pairs</u>	33
4.4.1	<u>Results</u>	33
4.4.2	<u>Discussion</u>	33
4.5	<u>Chemical Disorder</u>	37
4.5.1	<u>Results</u>	37
4.5.2	<u>Discussion</u>	37
5	<u>Conclusion</u>	41
6	<u>Bibliography</u>	42
7	<u>Appendix A: Program Lattice for Generating Model Lattices</u>	43
8	<u>Appendix B: RDF Fortran Programs</u>	48
9	<u>Appendix C: ADF Fortran Programs</u>	52
10	<u>Vita</u>	58

List of Figures

Figure 1: RDF for perfect and distorted simple cubic lattice	4
Figure 2: System of three atoms.	6
Figure 4: Inner ADF for perfect and distorted cubic lattice.	11
Figure 5: Outer ADF for perfect and distorted cubic lattice.	11
Figure 6: Unit cells for model systems.	14
Figure 7: RDF's for model systems.	15
Figure 8: Inner ADF's for model systems.	16
Figure 9: Outer ADF's for model systems.	17
Figure 10: Comparison of simulated systems with models.	20
Figure 11: Comparison of RDF's for liquid and amorphous FeTi	25
Figure 12: Comparison of FeTi and Si liquids. (Inner shell)	26
Figure 13: FeTi, NiTi, and Silicon Amorphous Systems (Inner)	26
Figure Figure 14: FeTi and Silicon liquids. (Outer Shell)	27
Figure 15: FeTi, NiTi, and Si amorphous systems. (Outer)	27
Comparison of Liquid and Amorphous FeTi (Outer Shell)	28
Comparison of Liquid and Amorphous FeTi (Inner Shell)	28
Figure 16: Increasing shell size for liquid FeTi. (Outer)	29
Figure 17: Increasing shell size for amorphous FeTi. (Outer)	30
Figure 18: Increasing shell size for liquid FeTi. (Inner)	31
Figure 19: Increasing shell size for amorphous FeTi. (Inner)	32
Figure 20: Increasing numbers of frenkel pairs. (FeTi)	35
Figure 21: Increasing numbers of frenkel pairs. (NiTi)	36
Figure 24: Increasing numbers of switches in FeTi.	39
Figure 25: Increasing number of switches in NiTi.	40

Abstract

An angular distribution function (ADF) was developed as a tool to be used in atomistic simulation. The ADF lets us easily distinguish between different lattice structures for systems of atoms by looking at the distribution of bond angles. The ADF is also a useful tool for determining the amount of order in a system. First, model structures of simple cubic, FCC, BCC, and diamond were developed. Then, the ADF was used on these model structures so that a set of model graphs of the ADF could be used to compare to real systems. Finally, simulated systems of ^{Fe-rich - Ti-rich} FeTi, NiTi, and Silicon were analyzed with the ADF. The known structures of the simulated systems all matched their model structures. The ADF was also able to distinguish between liquid and amorphous FeTi, but could not distinguish between liquid and amorphous Silicon. ^{Previously accomplished experiments to induce the transition from crystalline to amorphous with FeTi and NiTi (frenkel pairs and atom exchanges were introduced), were explored using the ADF and it was found that the ADF could be used to interpret the results accurately without the need to compute the structure factor.}

Developement of an Angular Distribution Function for the Study of Lattice Structures Used in Atomistic Simulation

1 Introduction

1.1 Background

Atomistic simulation is a method of investigating the properties of materials using a computer. One of the properties of concern is whether or not a system of atoms has a random distribution (amorphous), or whether they are in an ordered structure (crystal) and what kind of order (simple cubic, face centered cubic, etc...) they may have. The output of an atomistic simulation experiment includes the positions of all the atoms which makes a calculation of the structure possible. Clearly, an atom in a crystal lattice will see neighbors at an ordered distance from itself, and will see an ordered set of bond angles between itself and other atoms. Currently, the most common ways of determining the order of a system is by calculating a function called the radial distribution function (RDF), or by computing what is called the structure factor $S(\vec{k})$.

Neither of these make direct use of the bond angles of the system. The purpose of this research is to develop and explore a means of using the bond angles in a system to determine the type and degree of order.

1.2 General Approach

The approach to this research was first to write a program which calculates a function similar to the RDF but which includes bond angles. Once this was complete, model systems for simple cubic, face centered cubic (FCC), body centered cubic (BCC), and diamond were developed and the new function calculated for them. This would provide model ADF's for these systems so that an unknown system could be determined. Next, the function was calculated for the simulated systems of FeTi, NiTi, and Silicon. Finally, the results were analyzed to determine how the new function provides information on structure and order.

1.3 Sequence of Presentation

The presentation will start with a more detailed look at how structure and order are currently computed, and development of the new method. This is followed with the analysis of several model systems using this function. Finally, several simulated systems will be analyzed with the new function, and its validity will be determined.

2 Structure and Order

2.1 Introduction

As stated earlier, the most common means of determining structure and order are the radial distribution function, and the structure factor. Both of these will be explained, and then a new function which uses the bond angles will be introduced.

2.2 RDF

One of the most common methods of studying the structure of a system is to calculate the radial distribution function (RDF). The RDF is the ratio of a local number density in a spherical shell with the bulk number density. (1:54-55). (See Appendix A for Fortran programs which generate the RDF). A crystalline system would tend to have sharp peaks because the atoms are at well defined distances from each other. The peaks correspond to shells of neighbors surrounding each atom. If a system is disordered, then the peaks would become smoother. Figure 1 illustrates two RDF's; the top RDF is for a perfect simple cubic lattice while the bottom RDF is for a simple cubic lattice in which the atom positions have been distorted a random amount of up to 10% of the nearest neighbor distance.

A highly disordered system such as a liquid would show even smoother peaks which makes this technique very useful for distinguishing between highly-ordered and highly-disordered systems. Unfortunately, this technique washes out the angular distribution of the bonds

so it is more difficult to distinguish between systems which have some order but have a different structure. To compensate for this, a more complicated function called the structure factor is used.

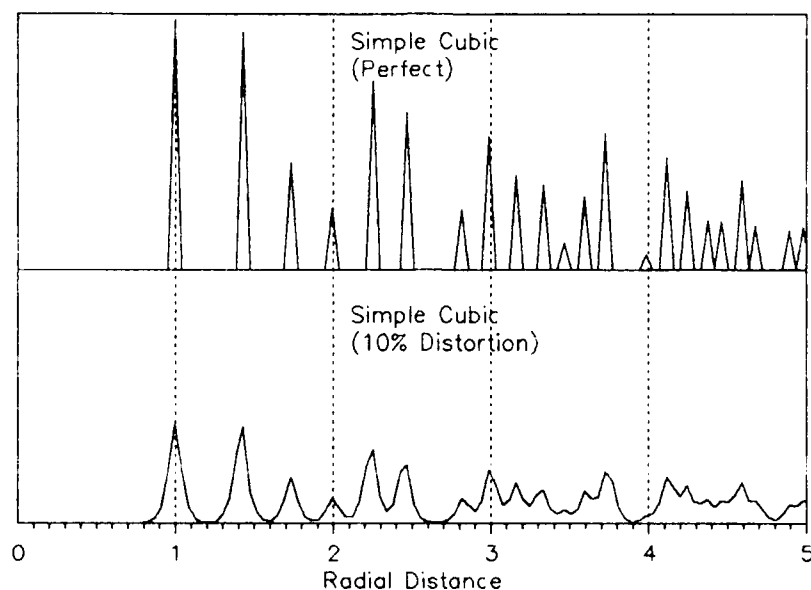


Figure 1. RDF for perfect and distorted simple cubic lattice

2.3 Structure Factor

The structure factor is used when the comparison of RDF's is not rigorous enough to distinguish between the structure of two systems. This is especially true for systems with a high amount of disorder because the only distinguishable peaks in the RDF will be associated with the nearest neighbors. But, if long-range order is present in one of the systems, the structure factor will be able to determine this.

The structure factor $S(\vec{k})$ is given by

$$S(\vec{k}) = \frac{1}{N} \left| \sum_{j=1}^N \exp(i\vec{k} \cdot \vec{r}_j) \right|^2 \quad (1)$$

where \vec{k} is a vector which represents a line of viewing through the system, N is the number of atoms, and \vec{r}_j is the vector position of atom j (2:3). The problem with this approach is that several viewing directions \vec{k} must be used in order to accurately analyze the system. In order to avoid this problem a simple solution would be to derive a function like the RDF but to incorporate the bond angles in such a manner that the structure of the system can be easily compared and distinguished from other systems. I have developed such a function which I will now refer to as an angular distribution function (ADF).

2.4 ADF

The ADF is built around the fact that every combination of three atoms forms a triangle with three bond angles. Therefore, the job of the ADF is to step through every combination of three atoms in a given system, determine the bond angles, and add those bond angles to a bin counter which represents some appropriate increment of 180 degrees of arc (such as one degree). After all bond angles are determined, each bin can be divided by the total number of bond angles so that each bin represents the probability that the bond angle between any three atoms will fall within that given increment of arc.

Although a true ADF should tell us the probability that the bond angle between any three atoms falls within a given increment of arc, it is

more informative to know bond angles between the first few nearest neighbors than with other bond angles, because that is where the structure of the system is evident. Therefore some kind of weighting must be applied to the data that is collected so that the importance of the nearest neighbor bond angles is not lost. Because of this inherent need for radial importance, the ADF will not give a true probability, but instead will give important trends in the angular distribution which is all that is really needed.

In order to come up with an appropriate weighting function the system of three atoms in Figure 2 must be considered.

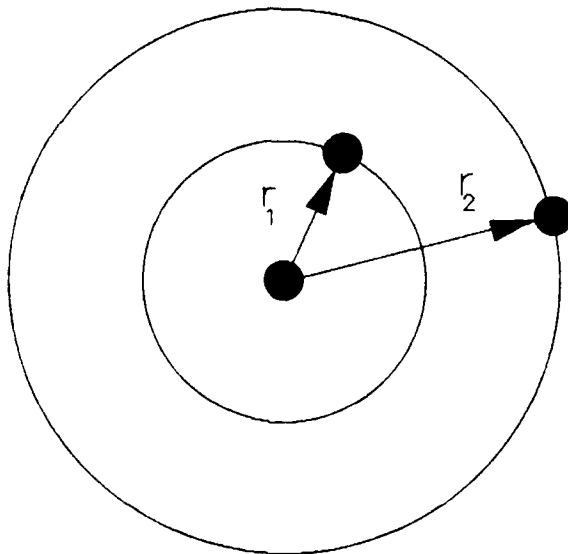


Figure 2. System of Three Atoms

The figure outlines two shells of radii r_1 and r_2 . If the central atom is assumed to be in the middle of a system of atoms, then the two shells enclose a number of atoms which all interact with a distribution of

bond angles. As the radius of either shell is increased, the number of atoms will increase as well as the number of corresponding bond angles. In fact, in an infinite system the number of atoms and their bond angles would also become infinite. Therefore, a weighting factor must be defined so that as either radius approaches infinity, the ADF will remain finite, or

$$\int_0^{\infty} \frac{4\pi r_1^2}{W(r_1)} dr_1 \int_0^{\infty} \frac{4\pi r_2^2}{W(r_2)} dr_2 = \text{Finite} \quad (2)$$

A simple solution for the weighting factor is then

$$W(r_1, r_2) = W(r_1)W(r_2) = r_1^4 r_2^4 \quad (3)$$

By weighting each angle by this factor the importance of the nearest neighbors will not be lost regardless of the size of the system. However, there are still other complications involved in developing an ADF.

Developing an angular distribution function is not as simple as developing an RDF because of the sheer number of combinations of angles in a system of atoms. An RDF which looks at all pair combinations of N atoms would have to compute for $N(N-1)/2$ combinations, but angles are formed with three atoms instead of two and there are three angles for each combination of three atoms. Therefore, the total number of angles in a system of N atoms would be $N(N-1)(N-2)/2$. This means that for

a typical system of 1000 atoms, the amount of computation for the ADF would be approximately three orders of magnitude greater than for the RDF. In real time it means that if the RDF takes 15 seconds to be computed, then the ADF would take over 4 hours to complete. This amount of time and computation is unacceptable if the ADF is to be a useful tool. Therefore, a way to eliminate bond angles and still retain useful information must be determined.

Since the bond angles of the nearest neighbors are what really define the structure of a system, then it would be advantageous to establish a cutoff shell of the first or second nearest neighbor distance (easily found with the RDF) and eliminate all bond angles between atoms that fall outside this shell. This eliminates a vast majority of bond angles and decreases the computational time by orders of magnitude. In addition, to insure that valuable information from outside this shell is not lost, another distribution must be computed which will be of bond angles between atom pairs from inside the cutoff shell with atoms from outside the shell. The first distribution will be called the inner ADF, and the latter will be called the outer ADF. Adding the outer ADF increased the computational time but it is still orders of magnitude faster than looking at all of the bond angles and loss of any useful information has been avoided. The algorithm for an ADF is presented in Figure 3 and the Fortran Program which will read Dynamo restart files can be found in Appendix B.

Using the same two simple cubic lattices that were used to illustrate the RDF in section 2.2, an inner and outer ADF can be made. However, the RDF must first be found in order to determine a shell cutoff radius. The first and second peaks in the RDF represent the first and second neighbors in the lattice, so in this instance a value of 1.2 is chosen which is midway between the two peaks. Figure 4 is an example of the inner ADF, and Figure 5 is an example of the outer ADF. Since the ADF uses a weighting factor, it can't be normalized and the specific height of a peak doesn't give any useful information, so the y-axis of the ADF's will not have any units labeled. The ADF has been scaled so that the highest peaks in each graph are the same height. This makes it easier to compare the ADF's since the resolution of the peaks will be approximately the same.

The broader peaks in the distorted lattice are indicative of the distortion, therefore the same sense of the amount of disorder that is available from the RDF is also present in the ADF. Also note, that the inner ADF is much easier to evaluate because of the number of peaks involved. For that reason, the inner ADF will always be used in the evaluations made in this paper unless the information is ambiguous or if more information is desired.

Given number of atoms NATOMS, atom positions R(), bin for outer angles
BIN1(), bin for inner angles BIN2()

```

For I=1 to NATOMS-1
  For J=I+1 to NATOMS
    Find  $|\vec{r}_{i,j}|$ 
    If  $|\vec{r}_{i,j}| \leq \text{SHELL}$ , Then
      For K=1 to NATOMS
        IF K  $\neq$  (I or J), Then
          Find  $|\vec{r}_{k,j}|$ 
          Find  $|\vec{r}_{i,k}|$ 
          WEIGHT1= $r_{i,j}^4 r_{k,j}^4$ 
          WEIGHT2= $r_{i,j}^4 r_{i,k}^4$ 
           $\theta_1 = \cos^{-1} \frac{\vec{r}_{i,j} \cdot \vec{r}_{k,j}}{|\vec{r}_{i,j}| |\vec{r}_{k,j}|}$ 
           $\theta_2 = \cos^{-1} \frac{\vec{r}_{i,j} \cdot \vec{r}_{i,k}}{|\vec{r}_{i,j}| |\vec{r}_{i,k}|}$ 
          If  $|\vec{r}_{k,j}| \geq \text{SHELL}$ , Then
            BIN1( $\theta_1$ ) = BIN1( $\theta_1$ ) + 1/WEIGHT1
          Else if K > J, Then
            BIN2( $\theta_1$ ) = BIN2( $\theta_1$ ) + 1/WEIGHT1
          End If
          If  $|\vec{r}_{i,k}| \geq \text{SHELL}$ , Then
            BIN1( $\theta_2$ ) = BIN1( $\theta_2$ ) + 1/WEIGHT2
          Else if K > J, Then
            BIN2( $\theta_2$ ) = BIN2( $\theta_2$ ) + 1/WEIGHT2
          End If
        End If
      End For
    End If
  End For
End For

```

Figure 3. Algorithm for ADF

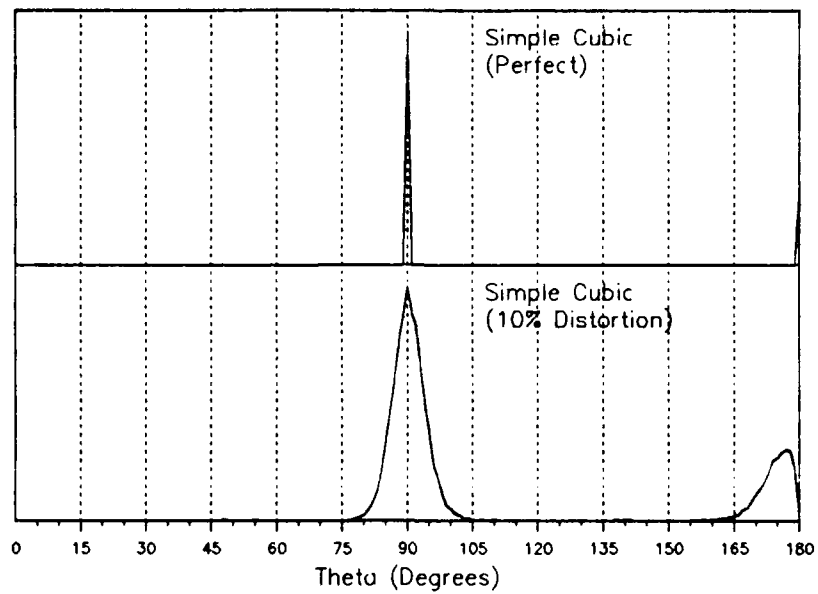


Figure 4. Inner ADF for Perfect and Distorted Cubic Lattice

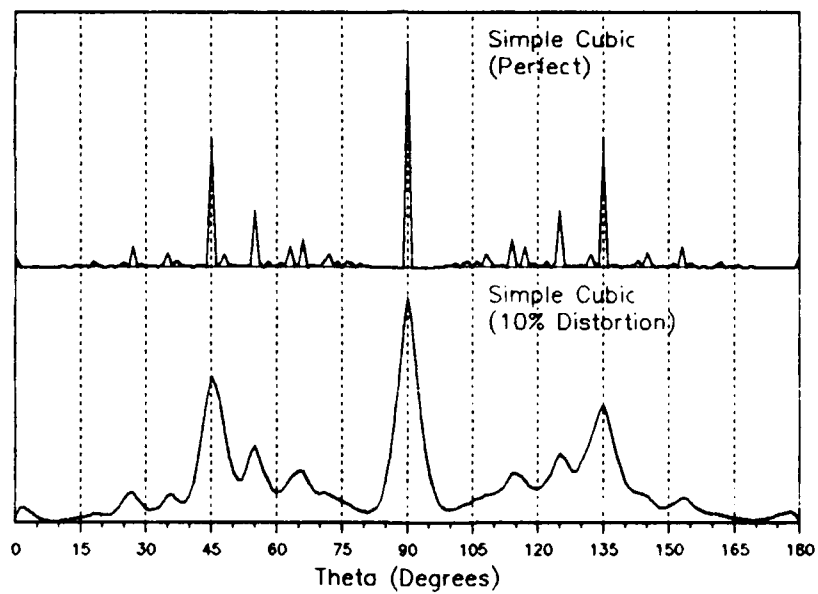


Figure 5. Outer ADF for Perfect and Distorted Cubic Lattice

3 Model Systems

3.1 Introduction

A set of model systems must be produced so that simulated systems can be compared. The model systems that will be developed are simple cubic, FCC, BCC, and diamond. Since most simulated systems will not be at a temperature of 0 K, then there will be some distortion in the lattice. Therefore, the models will be developed with a built in distortion. This is accomplished by using a lattice in which the atom positions have been randomly moved up to 10% of the nearest neighbor distance.

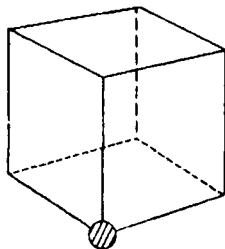
3.2 Results

The first step in creating the model system is to generate the lattices. This was accomplished by writing the program Lattice which is in Appendix C. The program generates a lattice of atoms by stacking unit cells of the appropriate type which are shown in Figure 6. After the lattices have been made it is necessary to find the RDF for each one so that the appropriate shell cutoff distance can be found. The shell cutoff thickness will be at the first or second nearest neighbor distance and is indicated in the RDF as the first "major" valley between peaks. Figure 7 shows the RDF's for each of the model systems. The cutoff thicknesses that will be used are:

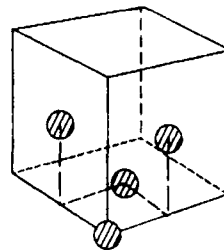
simple cubic: shell=1.2
FCC: shell=1.2
BCC: shell=1.4
diamond: shell=1.3

Note that for BCC the shell cutoff is between the second and third nearest neighbors. This is because the first and second nearest neighbors are so close in radial distance that it becomes impossible to distinguish separate peaks in the RDF as the disorder becomes greater. The results of the inner ADF can be found in Figure 8, and the results of the outer ADF are in Figure 9

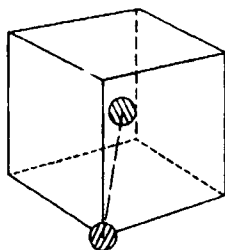
Simple Cubic



FCC



BCC



Diamond

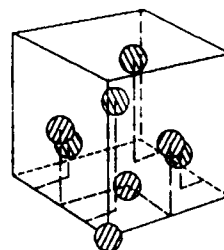


Figure 6. Unit Cells for Model Systems

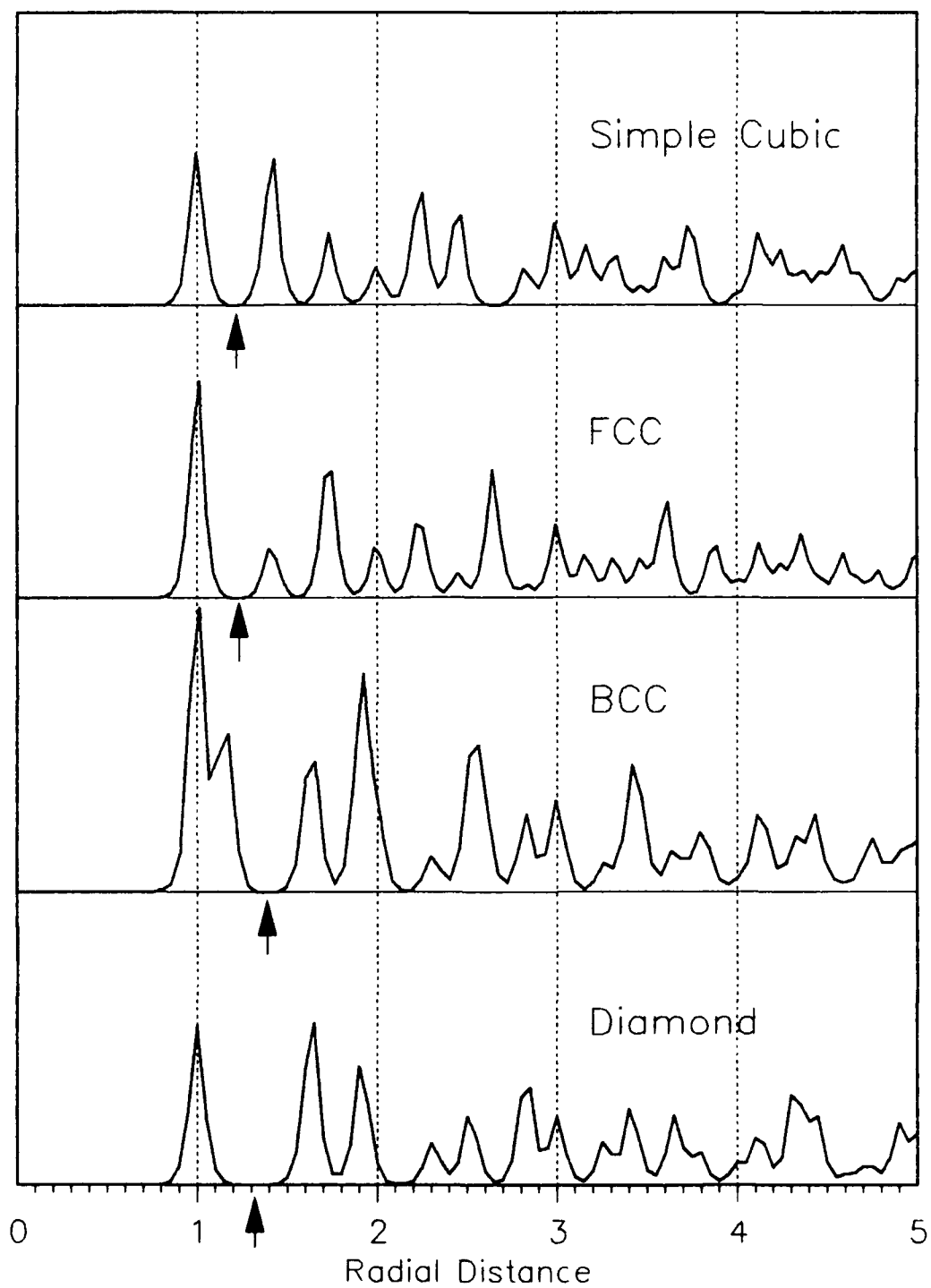


Figure 7. RDF's for Model Systems. Arrows Mark ADF Shells

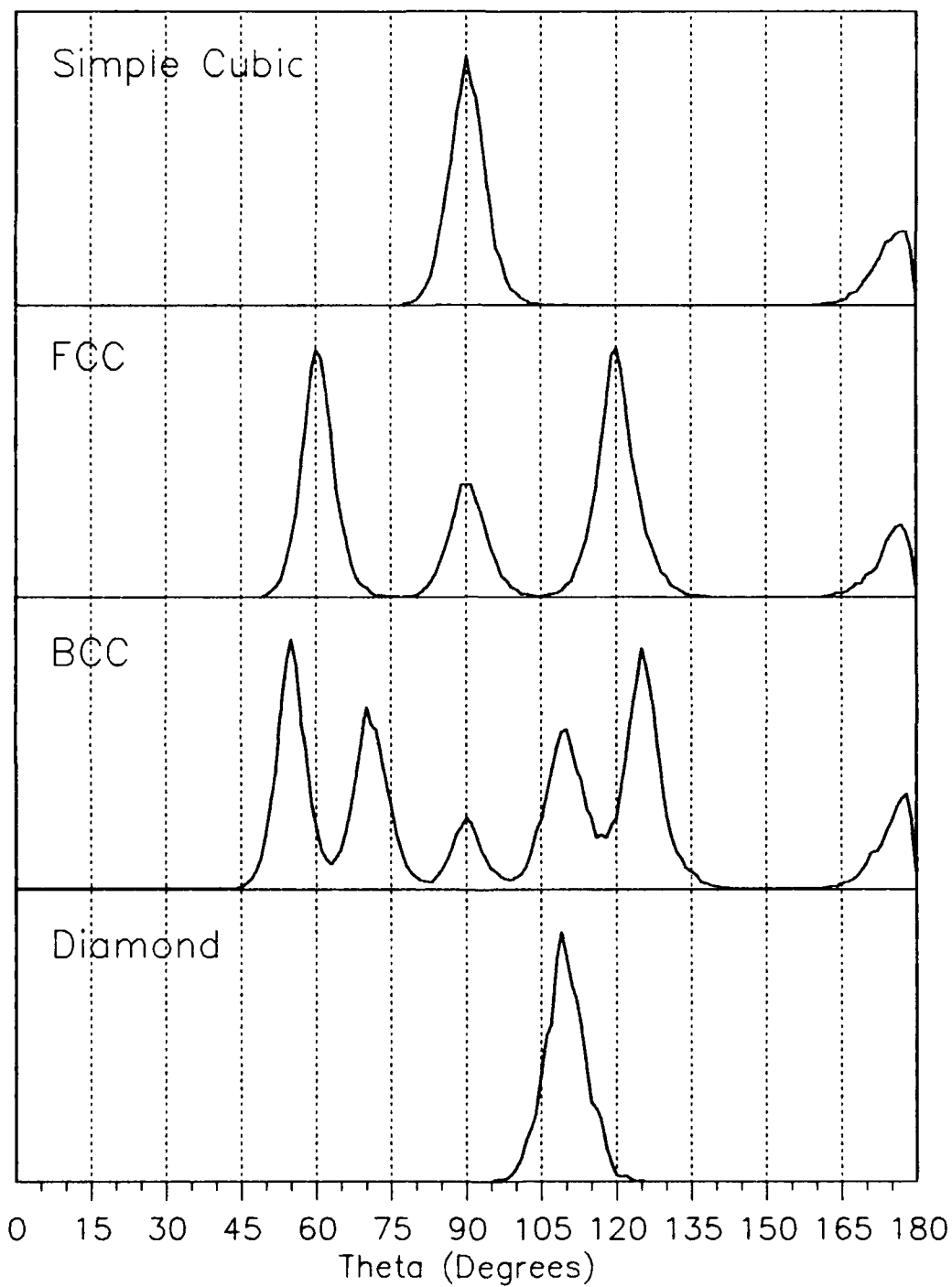


Figure 8. Inner ADF's for Model Systems

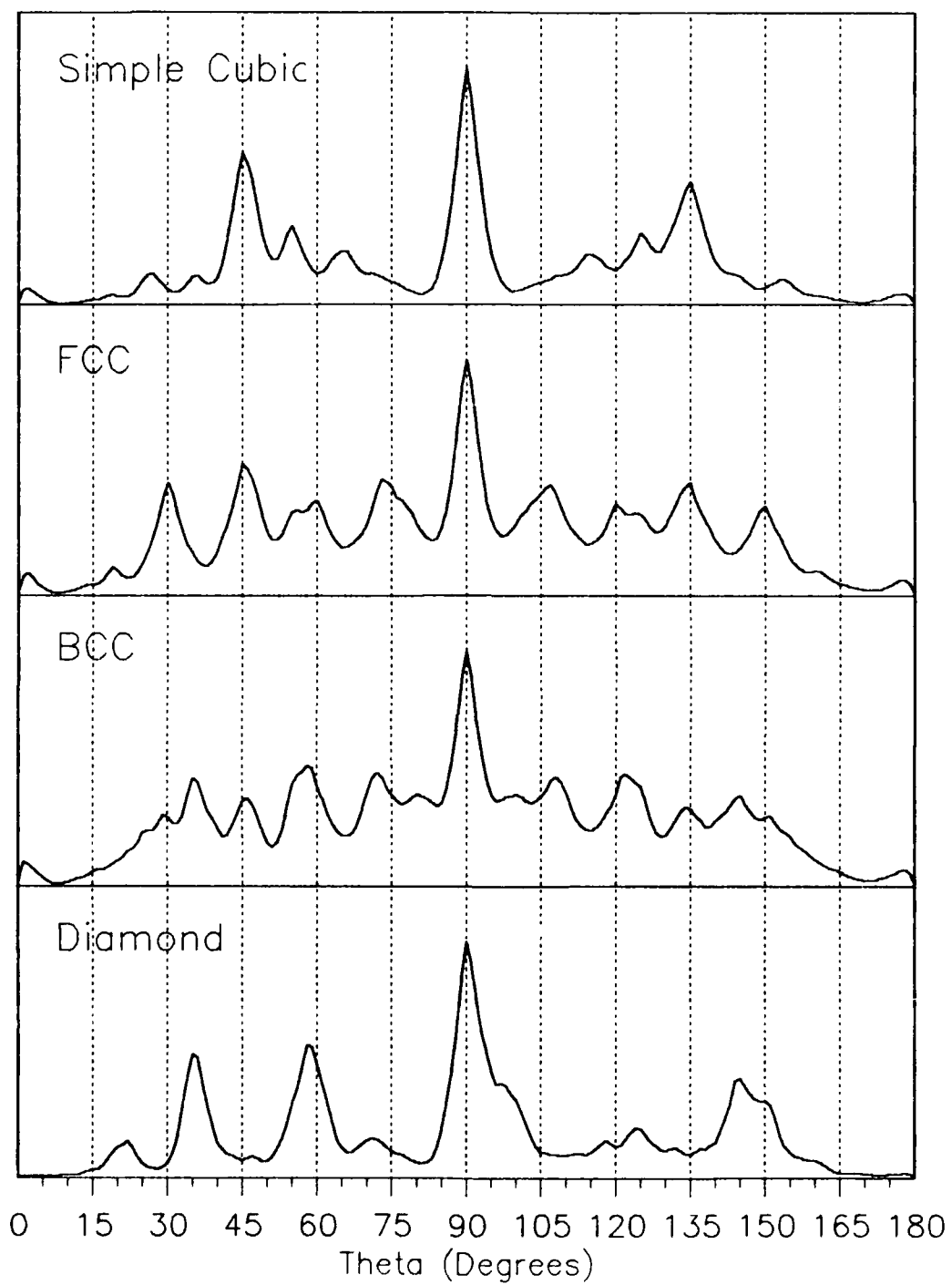


Figure 9. Outer ADF's for Model Systems

4 Simulated Systems

4.1 Introduction

Systems of FeTi, NiTi,(2) and Silicon (3) which were used by Sabo-chick and Lam for experiments with atomistic simulation will be analyzed using the ADF. Both FeTi and NiTi (1024 atoms in each system) have a structure which looks like BCC in their perfect states, while the Silicon (200 atoms in the system) has a diamond structure. First, each system will be compared to the model of its known crystal type to determine the validity of the model system. Next, the ADFs for the liquid and amorphous states of each system will be compared to see if the ADF can distinguish between the two states. Finally, for the FeTi and NiTi, the ADF will be used to look at systems which have switches and frenkel pairs, and then they will be compared with model systems and amorphous states.

4.2 Validation

4.2.1 Results

It is important to verify that the models produced accurately describe simulated systems. In order to do this model systems must be compared with simulated systems that have a known configuration. FeTi and NiTi are known to have a BCC configuration and Silicon has a diamond configuration. Compar-

ing the BCC and diamond models to these systems in Figure 10 it can be seen that the model systems do in fact accurately describe simulated systems.

4.2.2 Discussion

Since the models accurately describe simulated systems it is apparent that the ADF can be a useful tool for determining the structure of any ordered system. Note also that the simulated systems peaks are narrower than the Model peaks which were made with up to 10% distortion, therefore, the simulated systems have less than 10% distortion. If it was necessary to know the general amount of distortion in a system, a series of model ADF's with varying amounts of distortion could be made and used for comparison.

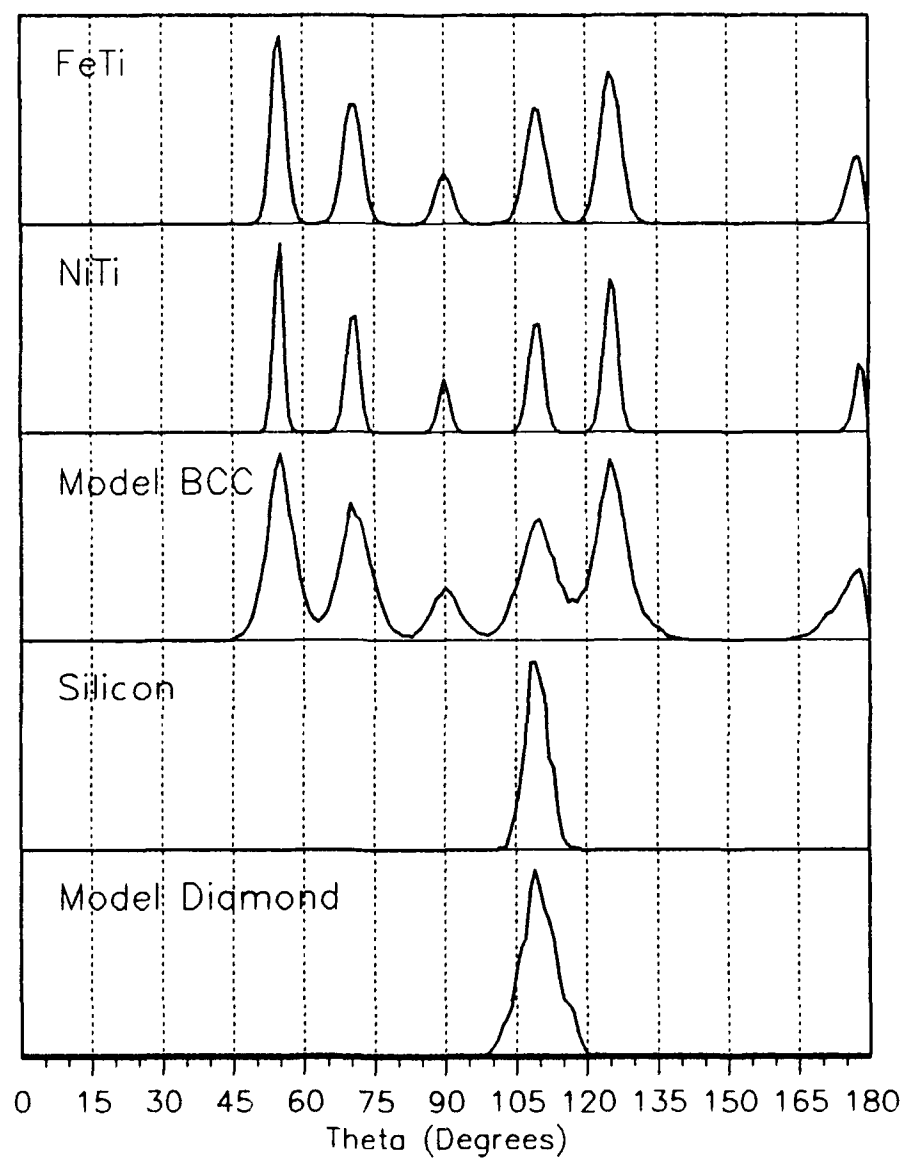


Figure 10. Comparison of Simulated Systems With Models

4.3 Liquid versus. Amorphous

4.3.1 Results

One of the important tools that this thesis hopes to provide is the ability to determine between a liquid and an amorphous solid. If the distinction is apparent, then the ADF will be able to establish where a liquid-solid interface exists by looking at blocks of atoms in a system. Both are characterized by disorder, so the RDF is not a good tool for distinguishing between the two (See Figure 11). Liquid systems for FeTi and Silicon, and amorphous systems for FeTi, NiTi, and Silicon were made available for this thesis. The Liquid Systems were made by heating a perfect lattice to 4000 K, and the amorphous systems were made by rapidly quenching the liquid to 160 K. Figure 12 compares the inner ADF's of the liquid and Figure 13 compares the inner ADF's of the amorphous states. Figure 14 and Figure 15 compare the outer ADF's of the liquid and amorphous states respectively. Finally, Figure 16 and Figure 17 show the outer and inner ADF's for comparisons of liquid and amorphous FeTi.

4.3.2 Discussion

The outer ADF's for the liquid and amorphous states are very similar. This is due to the fact that in a highly disordered system, the bond angles between a first or second

neighbor and neighbors farther out will become a smeared distribution. The Silicon system has a much rougher graph because the system only had approximately 200 atoms as compared to over 1000 for the FeTi and NiTi. The smallness of the Silicon system allows for gaps in the angular distribution. The similarity of the outer ADF's for liquid and amorphous systems appears to validate that the systems are in fact similar; namely that they are all very disordered.

The inner ADF's for the amorphous FeTi and NiTi are also very similar which is not surprising since they normally have the same structure, but the inner ADF's for the amorphous and liquid Silicon are both different from their FeTi and NiTi counterparts. This is not disturbing because there is no reason to expect these systems to have the same structure. What was disturbing was that the amorphous and liquid Silicon are both very similar to each other and in fact the ADF's appear to be just broadened peaks of the perfect diamond lattice. This may be because the potential function for Silicon which was used to create these lattices may not have created a true liquid which would tend toward a close-packed structure. Instead, the potential function seems to have kept the lattice in a diamond structure.

The FeTi and NiTi amorphous systems are supposed to be disordered, but they appear to have some sort of structure

which is indicated primarily by the peak at 60 degrees (in the inner ADF). To understand this, what has happened to the material must be understood. First, the metal is heated to melting, where many of the atomic bonds are ultimately broken. So that there is no longer any kind of crystalline structure. When the liquid is rapidly cooled (quenched), the atoms don't have time to return to any kind of orderly arrangement based on atomic bonding. Instead, they jostle for position in the amount of space that they have and atomic repulsion from each other takes precedence over atomic bonding. The resulting arrangement of atoms will be some kind of a close-packed structure such as HCP or FCC and is one that you might expect if you poured marbles into the bottom of a box. One of the key angles in HCP or FCC is 60 degrees, which is very evident in the amorphous ADF. The apparent lack of angles below about 40 degrees for both the liquid and amorphous system is due to the fact that the atoms of concern are within a defined shell which contains only first or second neighbors. Therefore, the bond angles can not be very small, because the atoms repel themselves from each other. The reason that the angles taper off going towards 180 degrees is that for a close-pack structure, atoms that are 180 degrees are going to be farther away than atoms at 60 or 120 degrees. Since the function is weighted against distance then those large angles

are weighted smaller.

There appears to be no conclusive difference between the liquid and amorphous outer ADF's. In fact, if the size of the shell is increased for both the liquid and amorphous FeTi, Figure 18 and Figure 19 show that eventually the outer ADF's are almost identical and show a general smear of all angles. On the other hand, Figure 20 and Figure 21 show that the inner ADF has maintained its distinction between liquid and amorphous. In fact, the inner ADF for the amorphous FeTi with increasing shell size shows a new peak begin to emerge between 35 and 40 degrees. This new peak corresponds to the second broad peak seen previously in the RDF for the amorphous FeTi. Therefore, the outer ADF is not a source of any real pertinent information for distinguishing between liquid and amorphous systems and can be removed as a useful tool. Instead, it might be advantageous to run more than one inner ADF at increasing shell sizes. This would provide all pertinent information and would take less computational time than running an outer ADF.

The primary difference between the FeTi and NiTi liquid and amorphous ADF's is that the amorphous systems have more well-defined peaks. This is enough to determine whether a system is liquid or quenched, but if the quenched system is heated up again, the peaks will broaden out and the distinction

between amorphous and liquid will be blurred unless the added heat allows the system to revert back into another type of crystal lattice. In that case the distinction between liquid and solid would be evident by structure and an interface could be established.

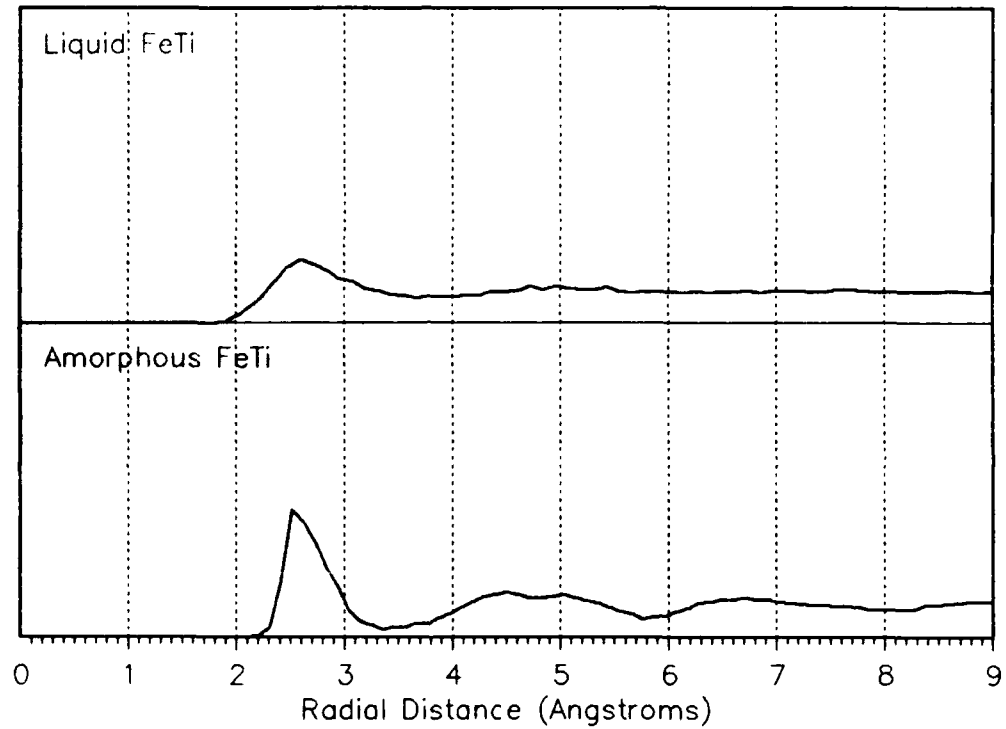


Figure 11. Comparison of RDF's for Liquid and Amorphous FeTi

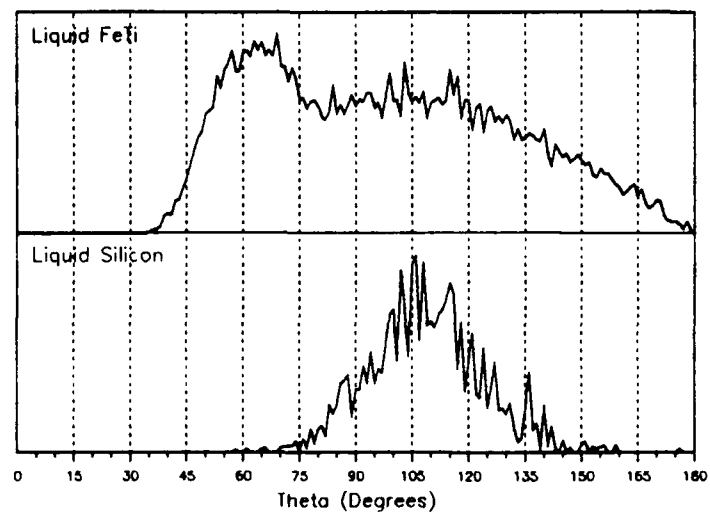


Figure 12. Comparison of FeTi and Silicon Liquids
(Inner Shell)

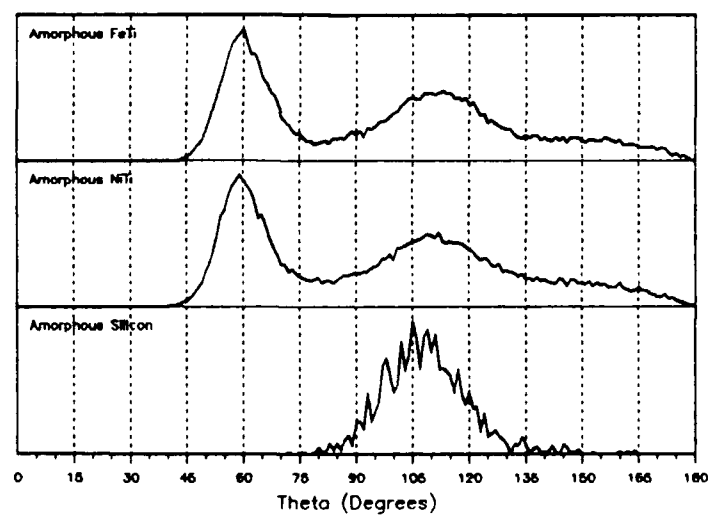


Figure 13. Comparison of FeTi, NiTi, and Silicon Amorphous
Systems (Inner Shell)

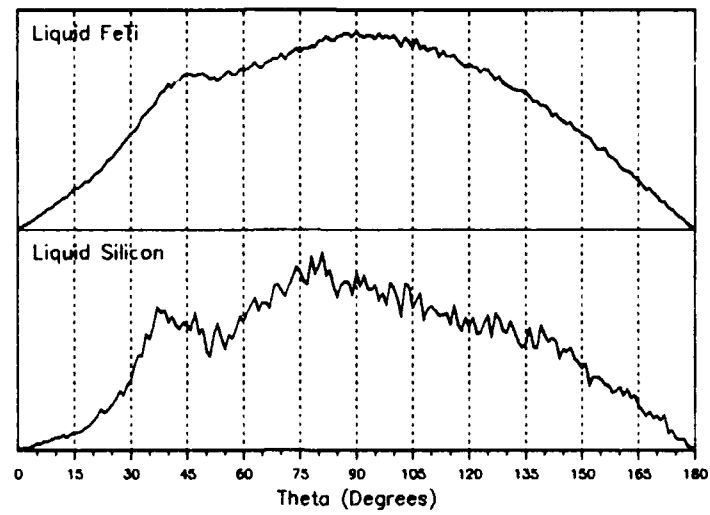


Figure 14. Comparison of FeTi and Silicon Liquids
(Outer Shell)

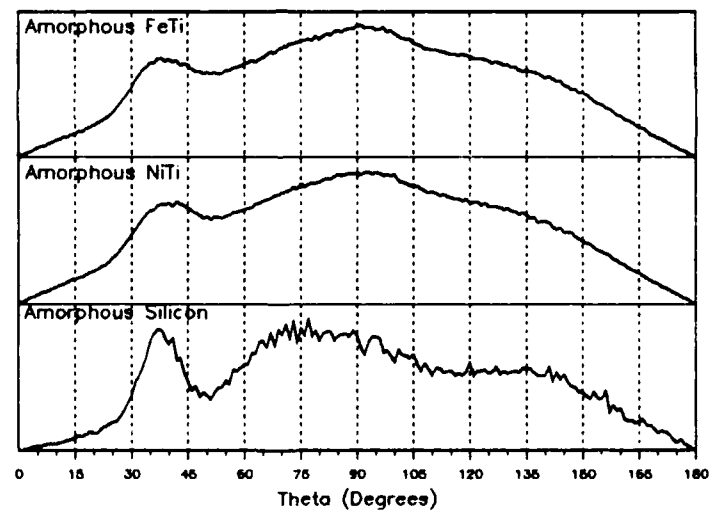
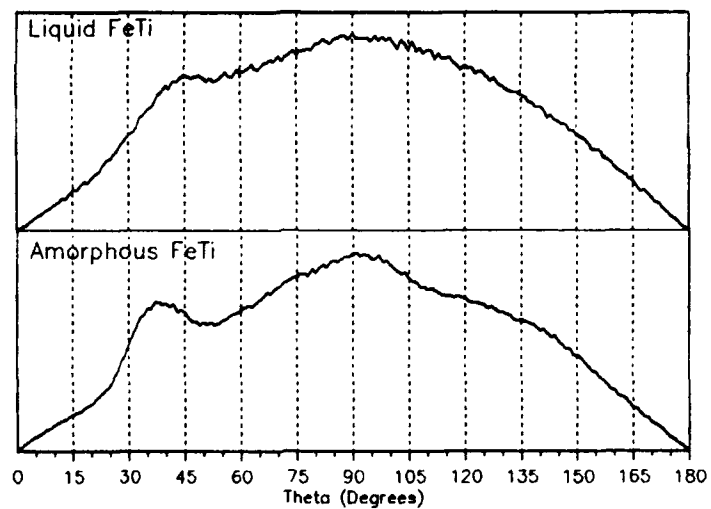
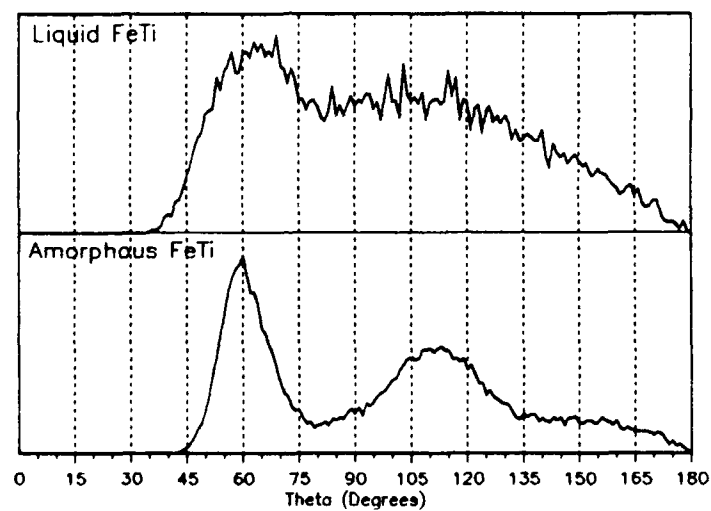


Figure 15. Comparison of FeTi, NiTi, and Silicon Amorphous
Systems (Outer Shell)



**Figure 16. Comparison of Liquid and Amorphous FeTi
(Outer Shell)**



**Figure 17. Comparison of Liquid and Amorphous FeTi
(Inner Shell)**

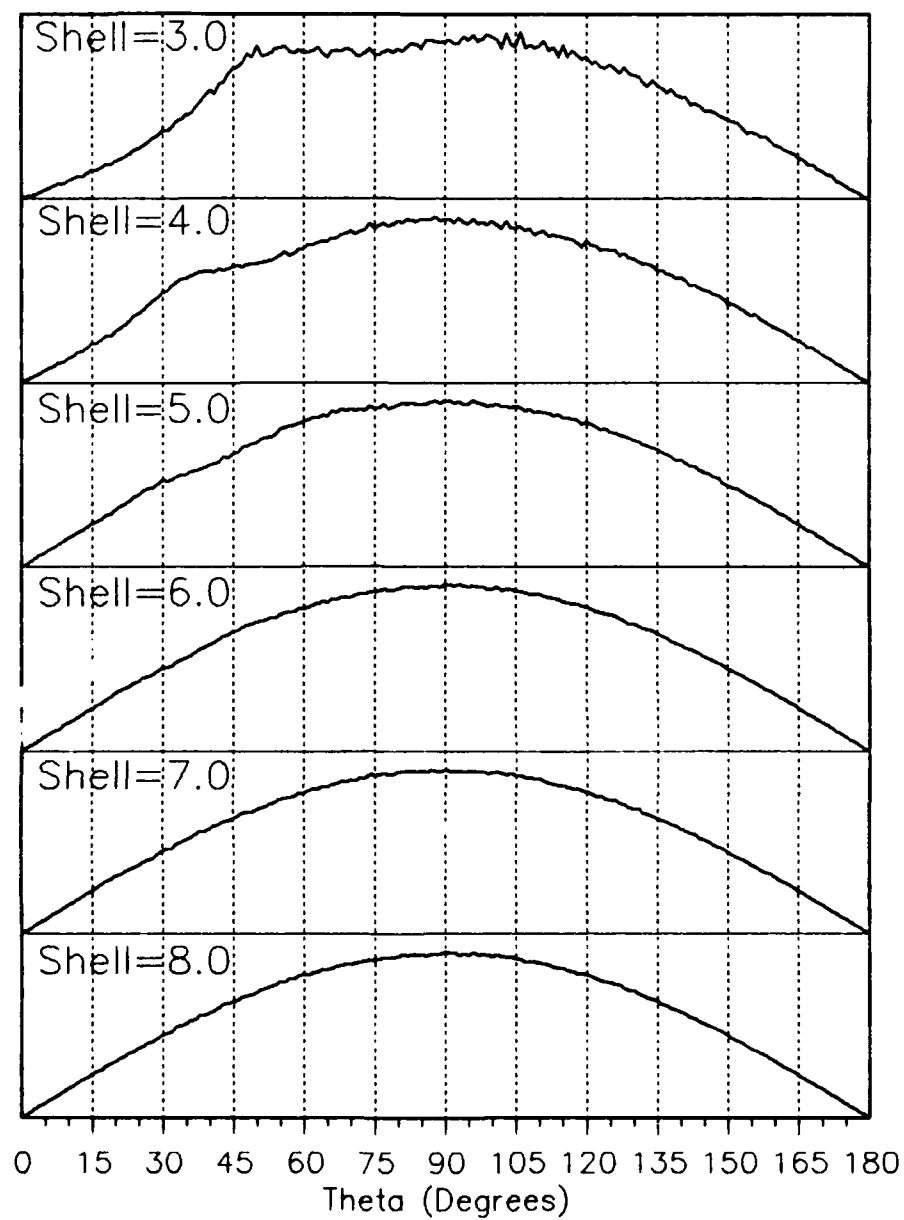
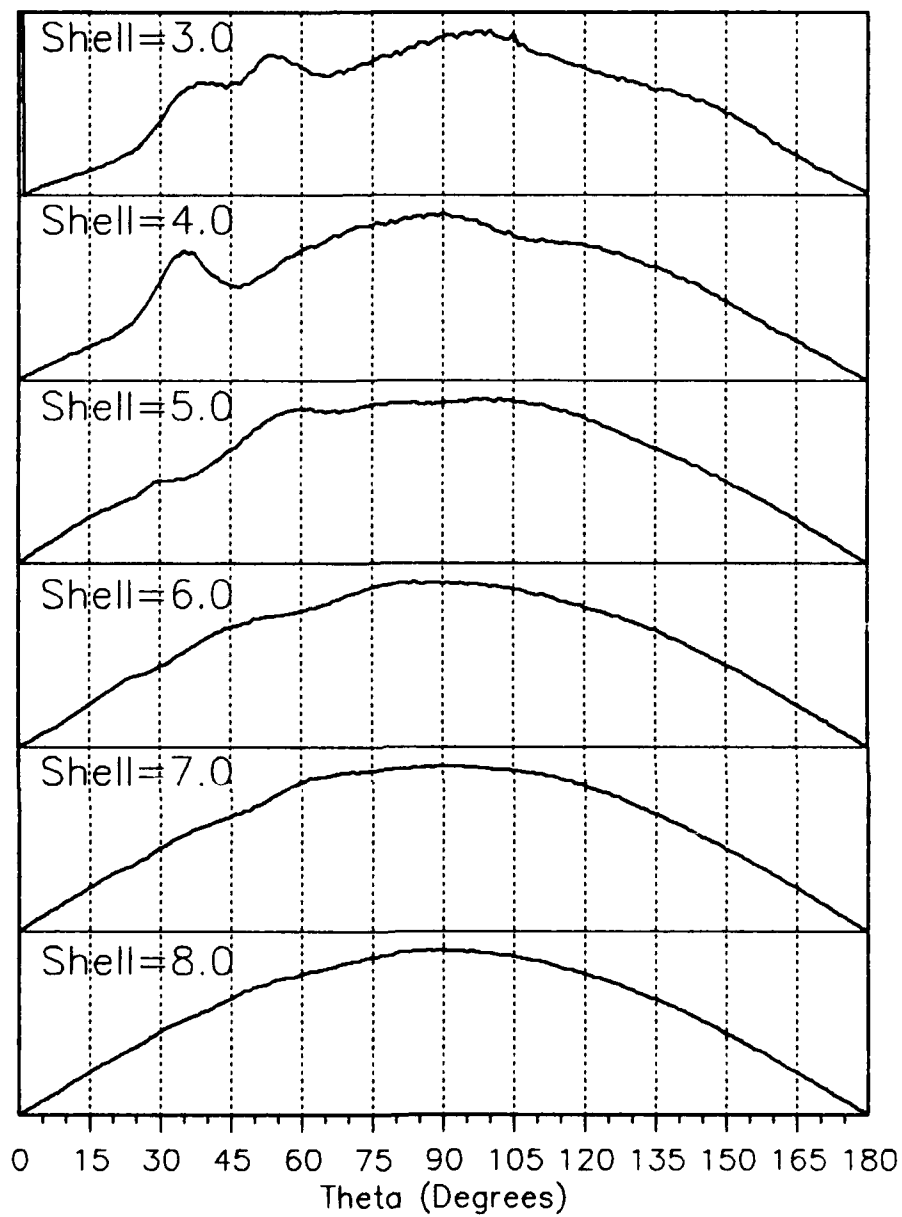
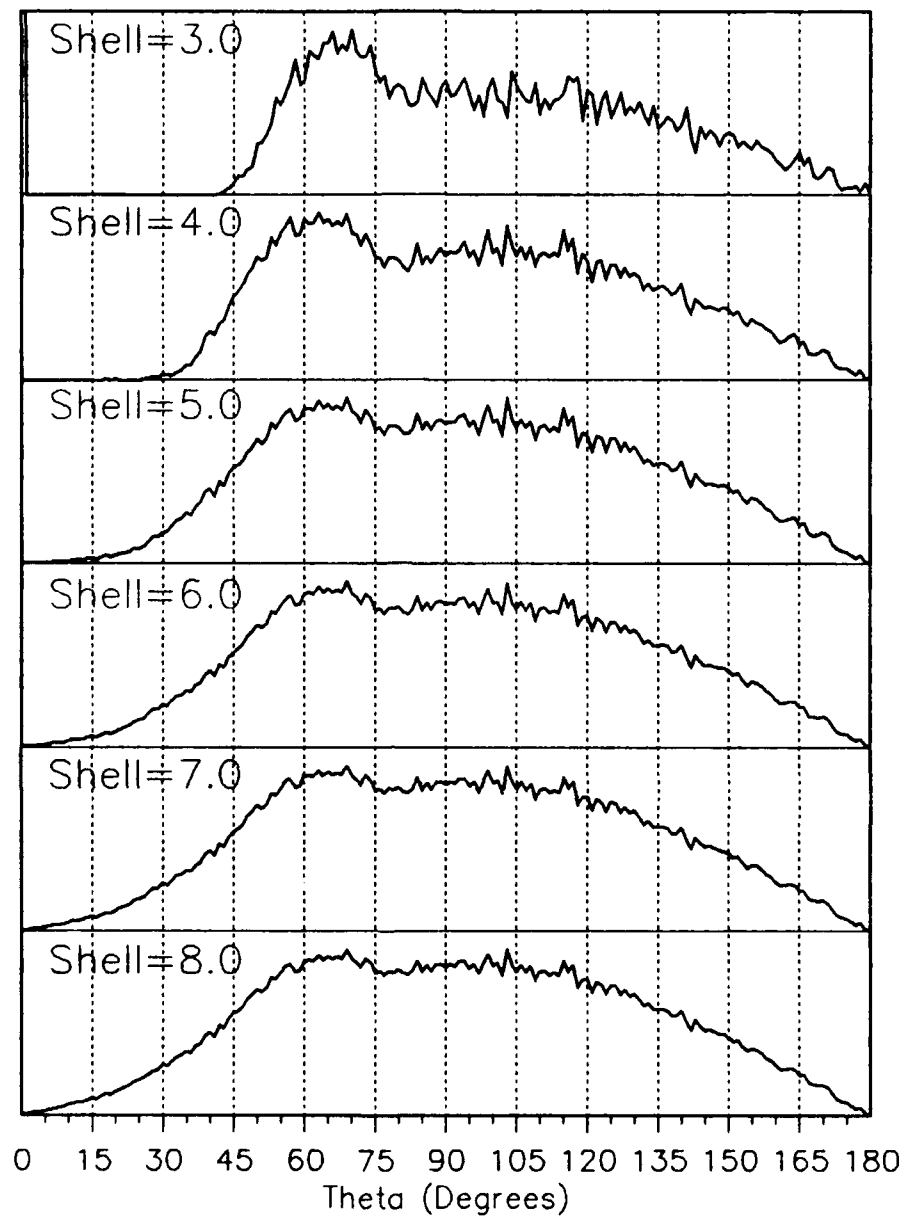


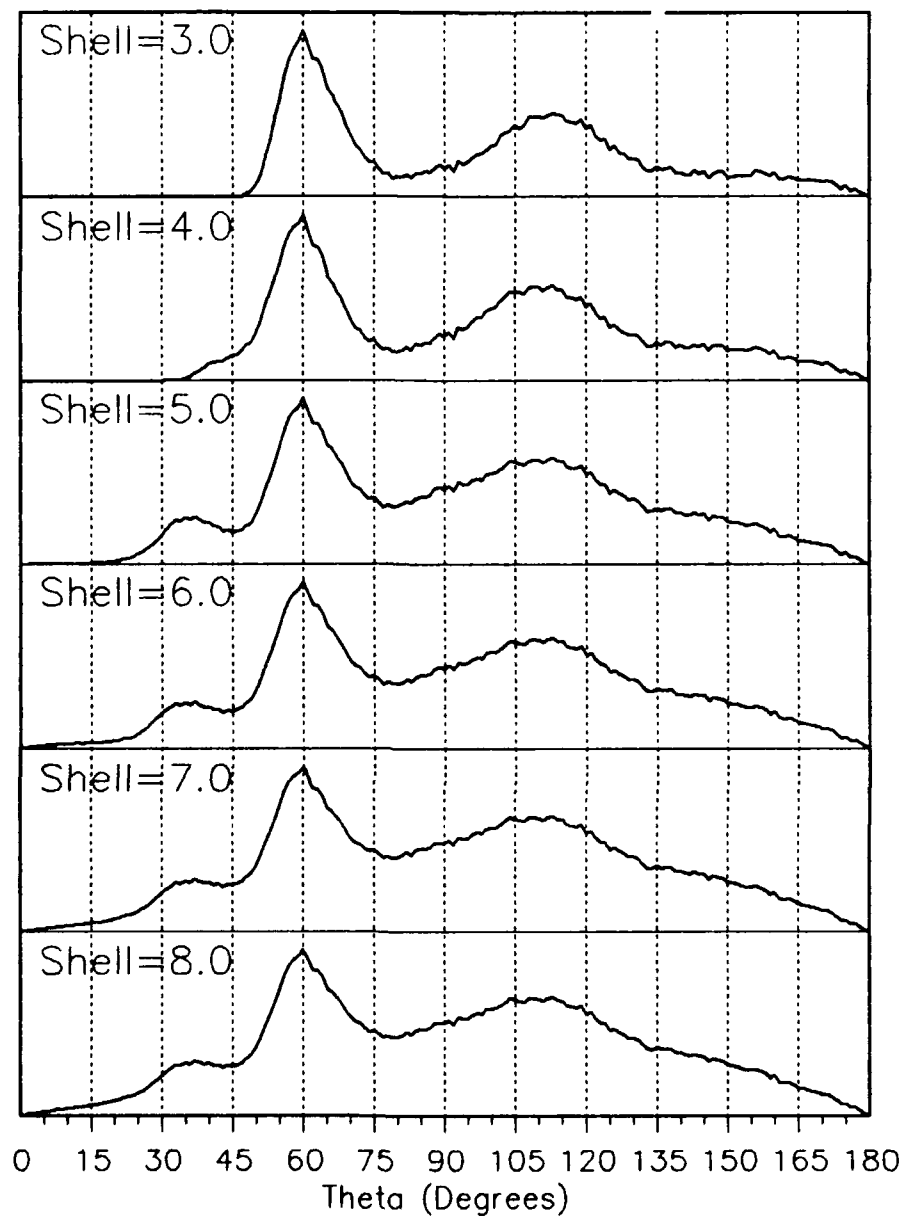
Figure 18. Increasing Shell Size for Liquid FeTi
(Outer Shell)



**Figure 19. Increasing Shell Size for Amorphous FeTi
(Outer Shell)**



**Figure 20. Increasing Shell Size for Liquid FeTi
(Inner Shell)**



**Figure 21. Increasing Shell Size for Amorphous FeTi
(Inner Shell)**

4.4 Frenkel Pairs

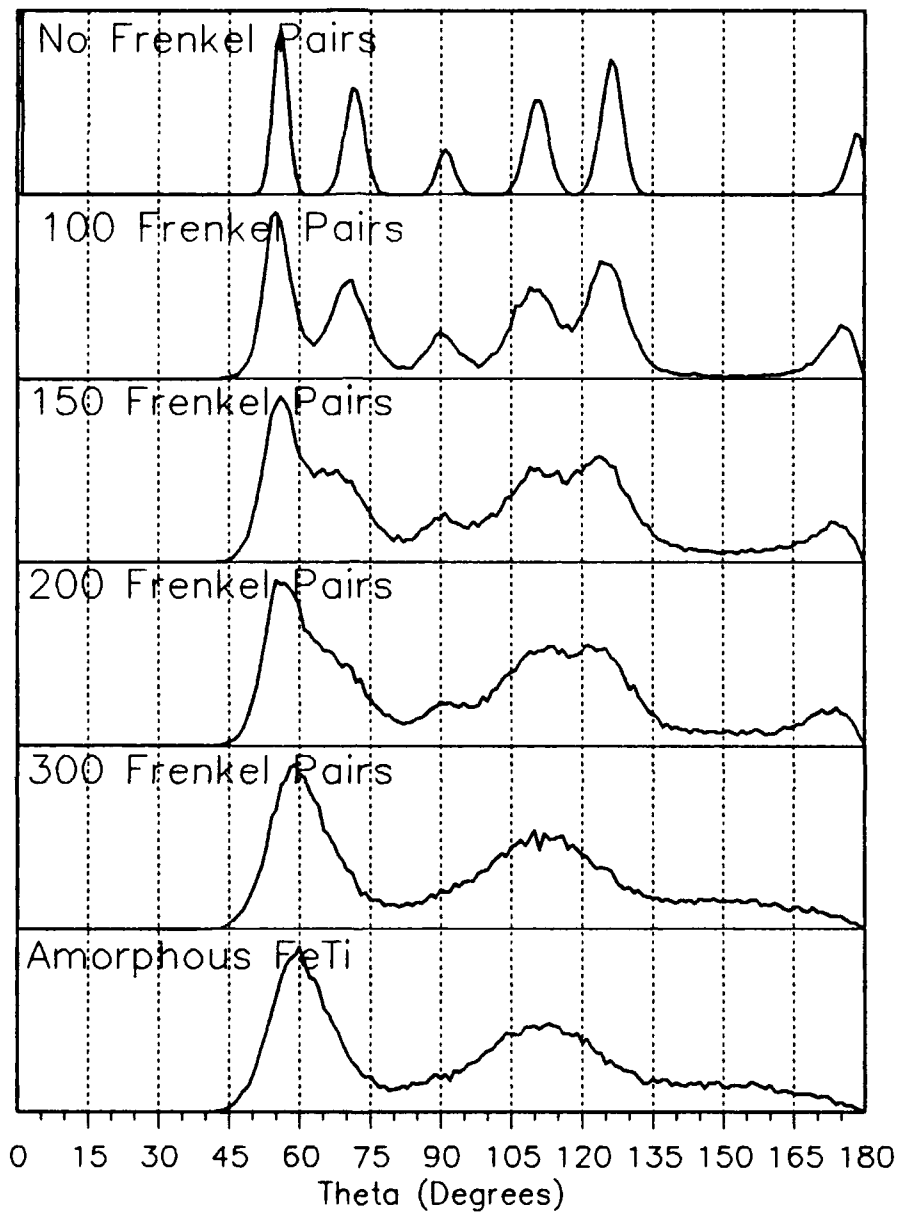
4.4.1 Results

Under electron irradiation, a crystal lattice develops point defects. When the normal site for an atom in a crystal lattice is empty, it is a point defect called a vacancy. When an atom sits in a position which is not one of the lattice sights, this is called an interstitial. When both of these defects occur together it is called a frenkel pair. It is known that an introduction of point defects can cause a system to change from being crystalline to being amorphous. Studies by Sabochick and Lam have been made with FeTi and NiTi (systems o 1024 atoms each) which show that the introduction of approximately 300 and 500 frenkel pairs respectively (2:2-4), are enough to induce an amorphous state.

4.4.2 Discussion

When an increasing number of frenkel pairs occur in both FeTi and NiTi the system changes its lattice structure from BCC to amorphous. Figure 22 shows the ADF's for an increasing number of frenkel pairs in FeTi and compares the ADF for 300 frenkel pairs with amorphous FeTi. It is apparent that the FeTi did change to an amorphous structure. Figure 23 shows similar reults for 500 frenkel pairs in NiTi. Therefore, the work of Sabochick and Lam (2) has been verified by the ADF.

These results have a significance greater than simply finding out whether the system becomes amorphous. The ADF now provides an easy method of pinpointing how many frenkel pairs it takes to make the system amorphous.



**Figure 22. ADF's for Increasing Numbers of Frenkel Pairs in
FeTi (System of 1024 atoms)**

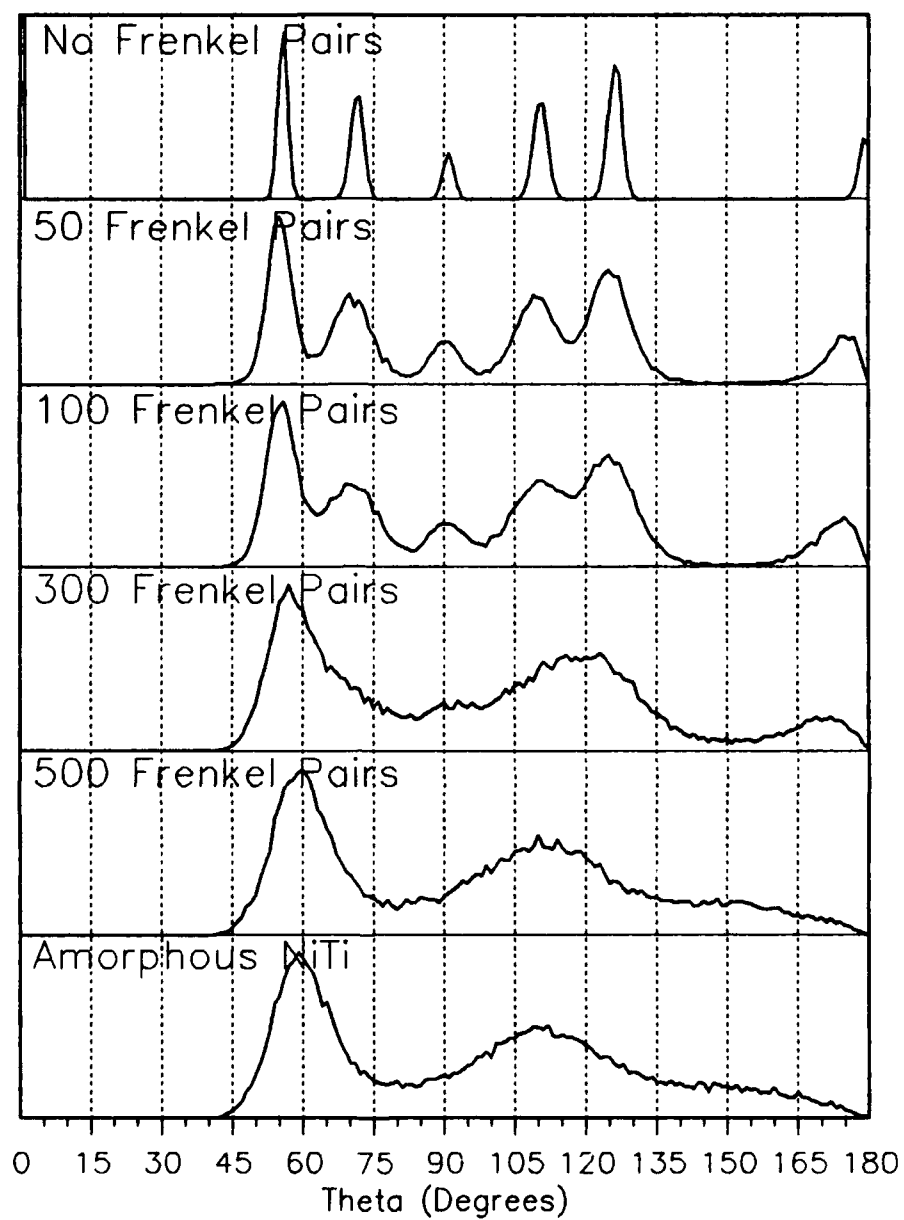


Figure 23. ADF's for Increasing Number of Frenkel Pairs in NiTi
(System of 1024 atoms)

4.5 Chemical Disorder

4.5.1 Results

In a crystalline lattice made up of more than one type of atom, each atom type holds a specific location in the unit cell. Under electron irradiation, the point defects which occur can cause chemical disordering whereby two different types of atoms switch positions. In some simulated compounds it has been found that enough of these random switches can produce amorphization of the material and are therefore the driving mechanism behind amorphization. FeTi is known not to behave in this manner; instead it undergoes a structure change from BCC to FCC. Work by Sabochick and Lam using the structure factor has shown that NiTi also fails to become amorphous with increasing number of switches (2:4), even though its RDF would appear to indicate otherwise.

4.5.2 Discussion

When an increasing number of switches occurs in FeTi the system changes its lattice structure from BCC to FCC. Comparing the ADF for 800 switches in FeTi with model FCC in Figure 24 it is apparent that the FeTi did change to an FCC structure.

Comparing the ADF for 1000 switches in NiTi with the amorphous NiTi in Figure 25, shows that there is an extra peak between 170 and 175 degrees and the other two peaks don't

quite match up in location with the amorphous system. This is a direct but simple verification of the work previously cited and shows that the switching of atoms is not the mechanism which produces amorphization of NiTi.

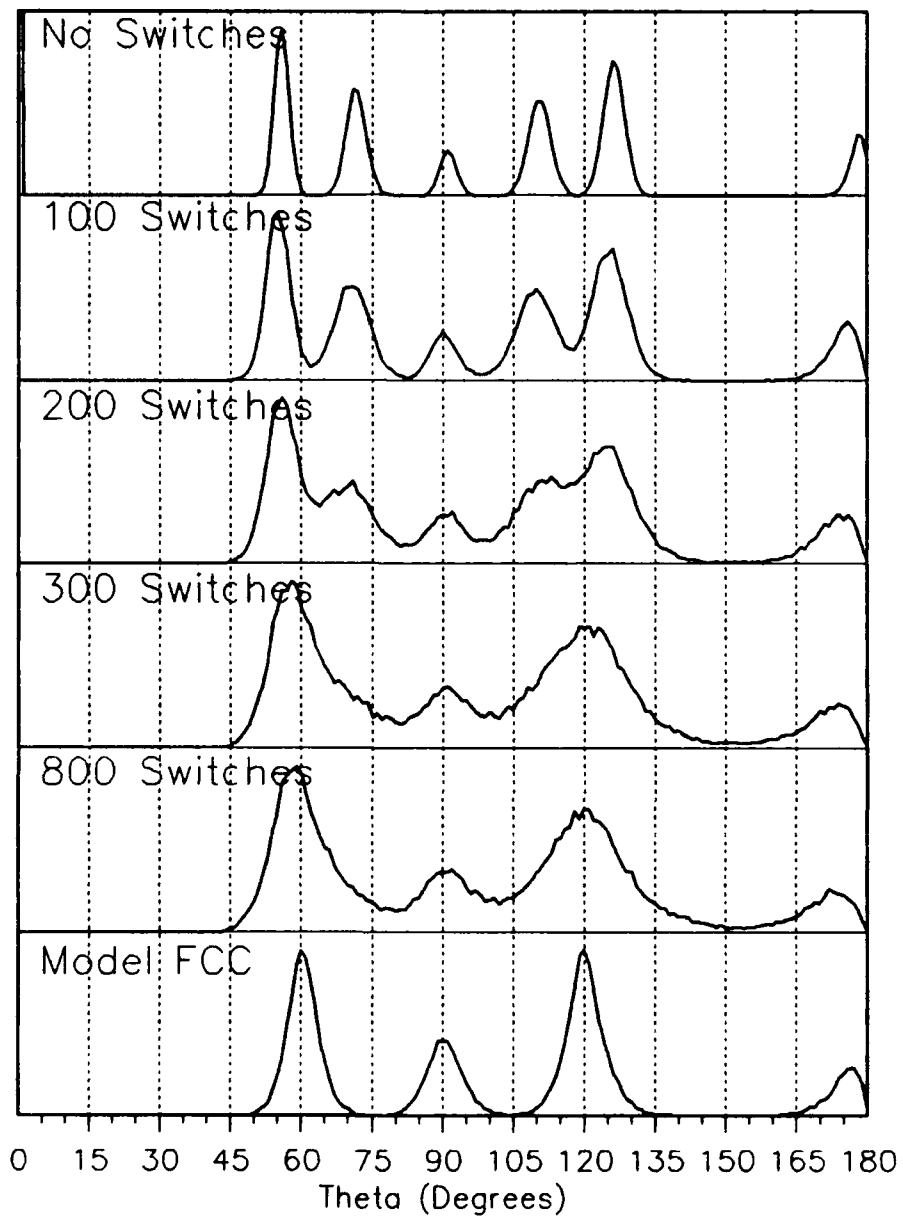


Figure 24. ADF's for Increasing Numbers of Switches in FeTi

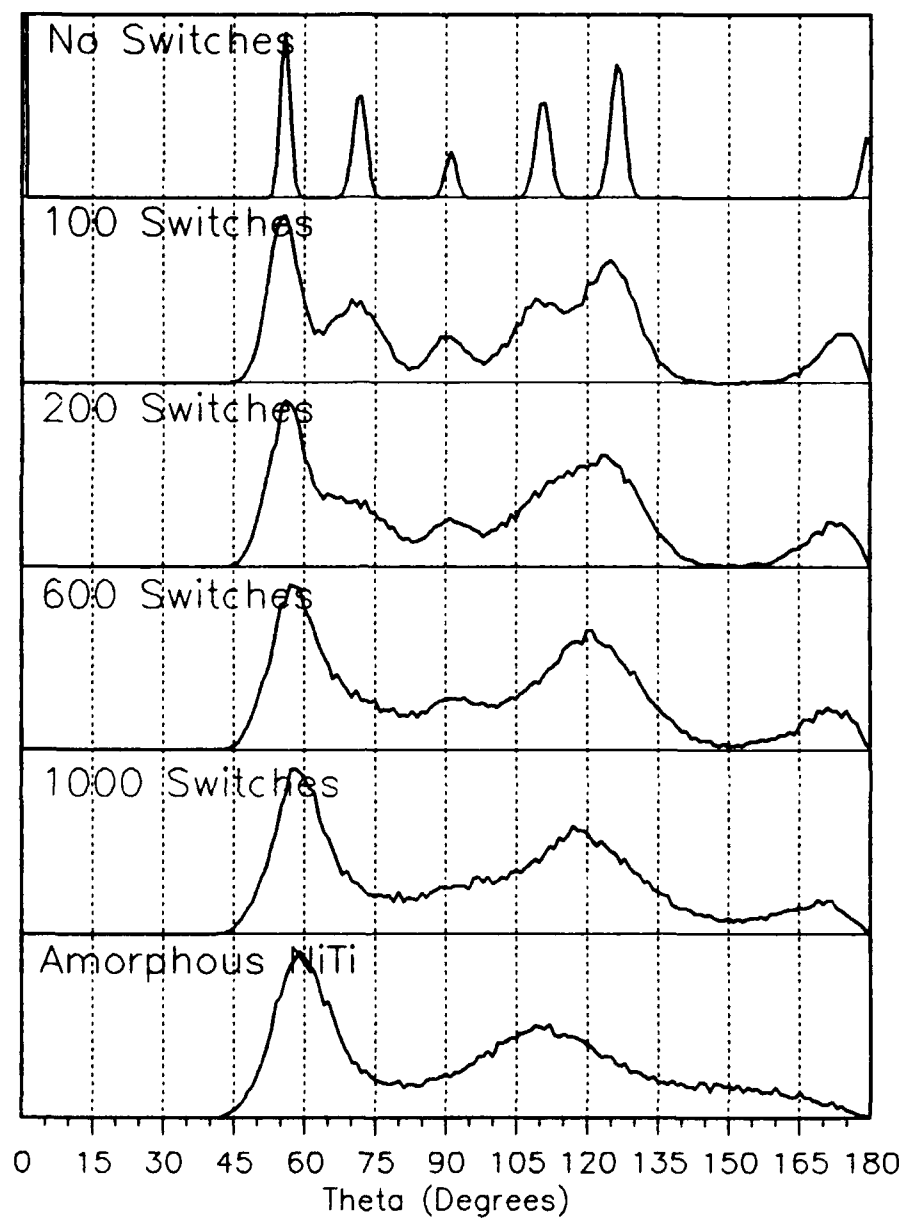


Figure 25. ADF's for Increasing Number of Switches in NiTi

5 Conclusion

A new function called an angular distribution function (ADF) has been developed for analyzing the structure of a system of atoms by finding the distribution of bond angles. Model systems of simple cubic, FCC, BCC, and diamond were developed so that they could be used for comparison with unknown structures. The model systems for BCC, FCC, and diamond were validated against atom systems of FeTi, NiTi, and Silicon which had those structures.

The ADF is useful in several ways. First, it is an indicator of the amount of order in a system in the same way that the RDF is an indicator; the broader the peaks are, the more disorder there is. Unlike the RDF, the ADF can reveal the type of structure that is inherent to the system. If a series of experiments are performed which lead to a change in structure (to include amorphicity) then the ADF will help to verify at what point in the experiment the change has occurred.

One thing that is not yet clear, is whether or not the ADF can be used in general to distinguish between a liquid system and a solid amorphous one. The studies here find that they are distinguishable for FeTi, but if the amorphous system were heated up, the peaks would broaden and the difference between the two would diminish. Further work should be accomplished in this area before it can be stated unequivocally that liquid and amorphous systems can be distinguished.

6 Bibliography

1. Haile, J. M. "A Primer on the Computer Simulation of Atomic Fluids by Molecular Dynamics." Report. Department of Chemical Engineering, Clemson University, Clemson, SC, November 1980.
2. Sabochick, Michael J. and Nghi Q. Lam. "Atomistic Simulation of Radiation-Induced Amorphization of the B2 Ordered Intermetallic Compound NiTi." Report to the Materials Research Society Fall Meeting, Symposium A: Surface Chemistry and Beam-Solid Interactions, Symposium F: Kinetics of Phase Transformations. Boston, MA, Nov. 26-30, 1990
3. Saobchick, Michael J., Associate Proffessor, Air Force Institute of Technology. Personal interview. Wright-Patterson AFB OH, 2 November, 1990.

7 Appendix A: Program Lattice for Generating Model Lattices

```
PROGRAM LATTICE
C
C  --DESCRIPTION--
C  Program Lattice makes model lattices by building a
C  series of unit cells to a set of dimensions which are
C  input by the user.
C
C  CHARACTER *15, OUTFILE
C
C  --OPEN FILES--
C
10  WRITE(6,*) 'ENTER OUTPUT FILENAME: '
    READ(5,20) OUTFILE
20  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=20, FILE=OUTFILE, STATUS='NEW', ERR=10)
C
C  --SET PARAMETERS--
C
C  Set the dimensions of the cell
C
    WRITE(6,*) 'ENTER NUMBER OF CELLS (X-DIRECTION): '
    READ(5,*)NXCELLS
    WRITE(6,*) 'ENTER NUMBER OF CELLS (Y-DIRECTION): '
    READ(5,*)NYCELLS
    WRITE(6,*) 'ENTER NUMBER OF CELLS (Z-DIRECTION): '
    READ(5,*)NZCELLS
    WRITE(6,*) 'ENTER CELL THICKNESS (ANGSTROMS): '
    READ(5,*)THICK
C
C  Determine the type of lattice for generation
C
    WRITE(6,*) 'ENTER # FOR TYPE OF CRYSTAL LATTICE'
    WRITE(6,*) 'SIMPLE CUBIC=1'
    WRITE(6,*) 'FCC=2'
    WRITE(6,*) 'BCC=3'
    WRITE(6,*) 'DIAMOND=4'
    READ(5,*)NTYPE
C
C  Enter a number < 1 and the atom positions will be
C  randomly distorted up to the entered fraction of
C  the nearest neighbor distance
C
    WRITE(6,*) 'ENTER FRACTION FOR RANDOM DISTORTION'
    READ(5,*)PERCENT
C
    PI=ACOS(-1.0)
```

```

C
DO 50, I=0,NXCELLS-1
  DO 40, J=0,NYCELLS-1
    DO 30, K=0,NZCELLS-1
C
C      Find the corner position of the unit cell
C
      X=I*THICK
      Y=J*THICK
      Z=K*THICK
C
C      Generate a Simple Cubic Lattice
C      (1 atom per cell)
C
      IF (NTYPE.EQ.1) THEN
        RMAX=PERCENT*THICK
        R1=RMAX*RAND(0)
        PHI1=PI*RAND(0)
        THETA1=2.0*PI*RAND(0)
        X1=X+R1*SIN(PHI1)*COS(THETA1)
        Y1=Y+R1*SIN(PHI1)*SIN(THETA1)
        Z1=Z+R1*COS(THETA1)
        WRITE(20,*)X1,Y1,Z1
      ELSE
C
C      Generate an FCC Lattice
C      (4 atoms per cell)
C
        IF(NTYPE.EQ.2) THEN
          RMAX=PERCENT*THICK/SQRT(2.0)
C
          R1=RMAX*RAND(0)
          PHI1=PI*RAND(0)
          THETA1=2.0*PI*RAND(0)
          X1=X+R1*SIN(PHI1)*COS(THETA1)
          Y1=Y+R1*SIN(PHI1)*SIN(THETA1)
          Z1=Z+R1*COS(THETA1)
C
          R2=RMAX*RAND(0)
          PHI2=PI*RAND(0)
          THETA2=2.0*PI*RAND(0)
          X2=X+0.5*THICK+R2*SIN(PHI2)*COS(THETA2)
          Y2=Y+R2*SIN(PHI2)*SIN(THETA2)
          Z2=Z+0.5*THICK+R2*COS(THETA2)
C
          R3=RMAX*RAND(0)
          PHI3=PI*RAND(0)
          THETA3=2.0*PI*RAND(0)
          X3=X+0.5*THICK+R3*SIN(PHI3)*COS(THETA3)

```

```

Y3=Y+0.5*THICK+R3*SIN(PHI3)*SIN(THETA3)
Z3=Z+R3*COS(THETA3)

```

C

```

R4=RMAX*RAND(0)
PHI4=PI*RAND(0)
THETA4=2.0*PI*RAND(0)
X4=X+R4*SIN(PHI4)*COS(THETA4)
Y4=Y+0.5*THICK+R4*SIN(PHI4)*SIN(THETA4)
Z4=Z+0.5*THICK+R4*COS(THETA4)

```

C

```

WRITE(20,*)X1,Y1,Z1
WRITE(20,*)X2,Y2,Z2
WRITE(20,*)X3,Y3,Z3
WRITE(20,*)X4,Y4,Z4

```

ELSE

C

C

Generate a BCC Lattice
(2 atoms per cell)

C

C

```

IF(NTYPE.EQ.3) THEN
  RMAX=PERCENT*THICK*SQRT(3.0)/2.0

```

C

```

R1=RMAX*RAND(0)
PHI1=PI*RAND(0)
THETA1=2.0*PI*RAND(0)
X1=X+R1*SIN(PHI1)*COS(THETA1)
Y1=Y+R1*SIN(PHI1)*SIN(THETA1)
Z1=Z+R1*COS(THETA1)

```

C

```

R2=RMAX*RAND(0)
PHI2=PI*RAND(0)
THETA2=2.0*PI*RAND(0)
X2=X+0.5*THICK+R2*SIN(PHI2)*COS(THETA2)
Y2=Y+0.5*THICK+R2*SIN(PHI2)*SIN(THETA2)
Z2=Z+0.5*THICK+R2*COS(THETA2)

```

C

```

WRITE(20,*)X1,Y1,Z1
WRITE(20,*)X2,Y2,Z2

```

ELSE

C

C

Generate a Diamond Lattice
(8 atoms per cell)

C

C

```

IF(NTYPE.EQ.4) THEN
  RMAX=PERCENT*THICK*SQRT(3.0)/4

```

C

```

R1=RMAX*RAND(0)
PHI1=PI*RAND(0)
THETA1=2.0*PI*RAND(0)

```

X1=X+R1*SIN(PHI1)*COS(THETA1)
Y1=Y+R1*SIN(PHI1)*SIN(THETA1)
Z1=Z+R1*COS(THETA1)

C

R2=RMAX*RAND(0)
PHI2=PI*RAND(0)
THETA2=2.0*PI*RAND(0)
X2=X+0.5*THICK+R2*SIN(PHI2)*COS(THETA2)
Y2=Y+R2*SIN(PHI2)*SIN(THETA2)
Z2=Z+0.5*THICK+R2*COS(THETA2)

C

R3=RMAX*RAND(0)
PHI3=PI*RAND(0)
THETA3=2.0*PI*RAND(0)
X3=X+0.5*THICK+R3*SIN(PHI3)*COS(THETA3)
Y3=Y+0.5*THICK+R3*SIN(PHI3)*SIN(THETA3)
Z3=Z+R3*COS(THETA3)

C

R4=RMAX*RAND(0)
PHI4=PI*RAND(0)
THETA4=2.0*PI*RAND(0)
X4=X+R4*SIN(PHI4)*COS(THETA4)
Y4=Y+0.5*THICK+R4*SIN(PHI4)*SIN(THETA4)
Z4=Z+0.5*THICK+R4*COS(THETA4)

C

R5=RMAX*RAND(0)
PHI5=PI*RAND(0)
THETA5=2.0*PI*RAND(0)
X5=X+0.25*THICK+R5*SIN(PHI5)*COS(THETA5)
Y5=Y+0.75*THICK+R5*SIN(PHI5)*SIN(THETA5)
Z5=Z+0.25*THICK+R5*COS(THETA5)

C

R6=RMAX*RAND(0)
PHI6=PI*RAND(0)
THETA6=2.0*PI*RAND(0)
X6=X+0.75*THICK+R6*SIN(PHI6)*COS(THETA6)
Y6=Y+0.25*THICK+R6*SIN(PHI6)*SIN(THETA6)
Z6=Z+0.25*THICK+R6*COS(THETA6)

C

R7=RMAX*RAND(0)
PHI7=PI*RAND(0)
THETA7=2.0*PI*RAND(0)
X7=X+0.25*THICK+R7*SIN(PHI7)*COS(THETA7)
Y7=Y+0.25*THICK+R7*SIN(PHI7)*SIN(THETA7)
Z7=Z+0.75*THICK+R7*COS(THETA7)

C

R8=RMAX*RAND(0)
PHI8=PI*RAND(0)
THETA8=2.0*PI*RAND(0)

C

```
X8=X+0.75*THICK+R8*SIN(PHI8)*COS(THETA8)
Y8=Y+0.75*THICK+R8*SIN(PHI8)*SIN(THETA8)
Z8=Z+0.75*THICK+R8*COS(THETA8)
```

```
WRITE(20,*)X1,Y1,Z1
WRITE(20,*)X2,Y2,Z2
WRITE(20,*)X3,Y3,Z3
WRITE(20,*)X4,Y4,Z4
WRITE(20,*)X5,Y5,Z5
WRITE(20,*)X6,Y6,Z6
WRITE(20,*)X7,Y7,Z7
WRITE(20,*)X8,Y8,Z8
```

```
ELSE
  STOP
END IF
END IF
```

```
END IF
```

```
END IF
```

```
30    CONTINUE
```

```
40    CONTINUE
```

```
50    CONTINUE
```

```
STOP
```

```
END
```

8 Appendix B: RDF Fortran Programs

```
PROGRAM RDF1
C
C   Program to find the RDF of a lattice generated by
C   the program Lattice
C
PARAMETER(MAXATMS=4000, MAXSIZE=200)
DIMENSION R(MAXATMS,3), BINDIST(0:MAXSIZE), WIDTH(3)
CHARACTER *15 OUTFILE,INPUT
REAL MAXDIST
C
C   --OPEN FILES--
C
10  WRITE(6,*) 'ENTER OUTPUT FILENAME: '
    READ(5,20)OUTFILE
20  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=20, FILE=OUTFILE, STATUS='NEW', ERR=10)
30  WRITE(6,*) 'ENTER INPUT FILENAME: '
    READ(5,40)INPUT
40  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=30, FILE=INPUT, STATUS='OLD', ERR=30)
C
C   --SET PARAMETERS--
C
WRITE(6,*) 'ENTER NUMBER OF ATOMS: '
READ(5,*) NATOMS
WRITE(6,*) 'ENTER THICKNES OF CELL (ANGSTROMS): '
READ(5,*) THICK
WRITE(6,*) 'ENTER EXPECTED FRACTION OF DISTORTION: '
READ(5,*) FRAC
WRITE(6,*) 'ENTER # OF CELLS ACROSS (X-DIRECTION): '
READ(5,*) NXCELLS
WRITE(6,*) 'ENTER # OF CELLS ACROSS (Y-DIRECTION): '
READ(5,*) NYCELLS
WRITE(6,*) 'ENTER # OF CELLS ACROSS (Z-DIRECTION): '
READ(5,*) NZCELLS
WRITE(6,*) 'ENTER NUMBER OF BINS FOR RDF: '
READ(5,*) NUMBINS
C
WIDTH(1)=THICK*NXCELLS
WIDTH(2)=THICK*NYCELLS
WIDTH(3)=THICK*NZCELLS
MAXDIST=SQRT(WIDTH(1)**2+WIDTH(2)**2+WIDTH(3)**2)*
*(1.0+FRAC)/2.0
BINSIZE=MAXDIST/NUMBINS
AVGDENS=NATOMS/(WIDTH(1)*WIDTH(2)*WIDTH(3))
```



```

        PI=ACOS(-1.0)
        NEXTRA=0
C
C      --INPUT POSITIONS--
DO 50 I=1,NATOMS
    READ(30,*)R(I,1),R(I,2),R(I,3)
50  CONTINUE
C
C      DO 100 I=1,NATOMS-1
        DO 80 J=I+1,NATOMS
            R2=0.0
            DO 60 K=1,3
                DR=R(I,K)-R(J,K)
                IF(ABS(DR).GT.(WIDTH(K)/2.0))THEN
                    DR=WIDTH(K)-ABS(DR)
                END IF
                R2=R2+DR**2
60      CONTINUE
            DIST=SQRT(R2)
            N=NINT(DIST/BINSIZE)
            IF(N.LE.NUMBINS) THEN
                BINDIST(N)=BINDIST(N)+1
            ELSE
                NEXTRA=NEXTRA+1
            END IF
80      CONTINUE
100     CONTINUE
C
C      --NORMALIZE--
DO 120 I=1,NUMBINS
    VOLUME=4.0/3.0*PI*((BINSIZE*I)**3-(BINSIZE*(I-1))**3)
    BINDIST(I)=(BINDIST(I)/VOLUME)/AVGDENS/NATOMS*2
120  CONTINUE
C
C      --OUTPUT--
DO 200 I=0,NUMBINS
    WRITE(20,*) I*BINSIZE, BINDIST(I)
200  CONTINUE
    WRITE(6,*)'NUMBER OF EXTRAS WHICH DID NOT FIT IN
    *BINS=',NEXTRA
C      --CLOSE FILES--

    ENDFILE(UNIT=20)
    CLOSE(UNIT=20)
C      --THE END--
    STOP
    END

```

```

PROGRAM RDF2
C
C   Program to find the RDF of a lattice generated by
C   the program Dynamo
C
PARAMETER(MAXATMS=4000, MAXSIZE=200)
DIMENSION R(MAXATMS,3), BINDIST(0:MAXSIZE), WIDTH(3)
CHARACTER *15 OUTFILE,INPUT
CHARACTER *80 STRING1
REAL MAXDIST
C
C   --OPEN FILES--
C
10  WRITE(6,*) 'ENTER OUTPUT FILENAME: '
    READ(5,20)OUTFILE
20  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=20, FILE=OUTFILE, STATUS='NEW', ERR=10)
30  WRITE(6,*) 'ENTER INPUT FILENAME: '
    READ(5,40)INPUT
40  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=30, FILE=INPUT, STATUS='OLD', ERR=30)
C
C   --SET PARAMETERS--
C
WRITE(6,*) 'ENTER NUMBER OF BINS FOR RDF: '
READ(5,*) NUMBINS
C
C   --INPUT DATA FROM FILE--
READ(30,*)STRING1
READ(30,*)NATOMS,NTYPES,TRASH1
READ(30,*)WIDTH(1),WIDTH(2),WIDTH(3)
READ(30,*)TRASH2,TRASH3,TRASH4
DO 45 I=1,NTYPES
    READ(30,*)TRASH5,TRASH6
45  CONTINUE
DO 50 I=1,NATOMS
    READ(30,*)R(I,1),R(I,2),R(I,3)
    READ(30,*)TRASH7,TRASH8,TRASH9
    READ(30,*)TRASH10
50  CONTINUE
C
C
MAXDIST=SQRT(WIDTH(1)**2+WIDTH(2)**2+WIDTH(3)**2)/2.0
BINSIZE=MAXDIST/NUMBINS
AVGDENS=NATOMS/(WIDTH(1)*WIDTH(2)*WIDTH(3))
PI=ACOS(-1.0)
C

```

```

DO 100 I=1,NATOMS-1
  DO 80 J=I+1,NATOMS
    R2=0.0
    DO 60 K=1,3
      DR=R(I,K)-R(J,K)
      IF(ABS(DR).GT.(WIDTH(K)/2.0))THEN
        DR=WIDTH(K)-ABS(DR)
      END IF
      R2=R2+DR**2
60    CONTINUE
      DIST=SQRT(R2)
      N=NINT(DIST/BINSIZE)
      BINDIST(N)=BINDIST(N)+1
80    CONTINUE
100  CONTINUE
C
C    --NORMALIZE--
DO 120 I=1,NUMBINS
  VOLUME=4.0/3.0*PI*((BINSIZE*I)**3-(BINSIZE*(I-1))**3)
  BINDIST(I)=(BINDIST(I)/VOLUME)/AVGDENS/NATOMS*2
120  CONTINUE
C
C    --OUTPUT--
DO 200 I=0,NUMBINS
  WRITE(20,*) I*BINSIZE, BINDIST(I)
200  CONTINUE
C    --CLOSE FILES--

  ENDFILE(UNIT=20)
  CLOSE(UNIT=20)
C    --THE END--
  STOP
  END

```

9 Appendix C: ADF Fortran Programs

```
      PROGRAM ADF1
C
C      Program to find the ADF of a lattice generated by
C      the program Lattice
C
      PARAMETER(MAXATMS=4000)
      DIMENSION R(MAXATMS,3), BIN1(0:180), BIN2(0:180),
      *A(3), B(3), C(3), WIDTH(3)
      CHARACTER *15 OUTFILE,INPUT
      REAL MAGA2, MAGB2, MAGC2, MAGA, MAGB, MAGC
      INTEGER U, V
C
C      --OPEN FILES--
C
10  WRITE(6,*) 'ENTER OUTPUT FILENAME: '
    READ(5,20)OUTFILE
20  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=20, FILE=OUTFILE, STATUS='NEW', ERR=10)
30  WRITE(6,*) 'ENTER INPUT FILENAME: '
    READ(5,40)INPUT
40  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=30, FILE=INPUT, STATUS='OLD', ERR=30)
C
C      --SET PARAMETERS--
C
    WRITE(6,*) 'ENTER NUMBER OF ATOMS: '
    READ(5,*) NATOMS
    WRITE(6,*) 'ENTER THICKNESS OF CELL:'
    READ(5,*) THICK
    WRITE(6,*) 'ENTER # OF CELLS THICK (X-DIRECTION): '
    READ(5,*) NXCELLS
    WRITE(6,*) 'ENTER # OF CELLS THICK (Y-DIRECTION): '
    READ(5,*) NYCELLS
    WRITE(6,*) 'ENTER # OF CELLS THICK (Z-DIRECTION): '
    READ(5,*) NZCELLS
    WRITE(6,*) 'ENTER RADIAL SUPPRESSION CUTOFF THICKNESS'
    READ(5,*) SHELL
C
    WIDTH(1)=NXCELLS*THICK
    WIDTH(2)=NYCELLS*THICK
    WIDTH(3)=NZCELLS*THICK
    PI=ACOS(-1.0)
C
C      --INPUT POSITIONS--
    DO 50 I=1,NATOMS
```

```

      READ(30,*)R(I,1),R(I,2),R(I,3)
50  CONTINUE
C
C
      DO 100 I=1,NATOMS-1
        DO 90 J=I+1,NATOMS
          MAGA2=0.0
          DO 80 L=1,3
            A(L)=R(J,L)-R(I,L)
            IF(ABS(A(L)).GT.(WIDTH(L)/2))THEN
              IF(A(L).GT.0.0)THEN
                A(L)=A(L)-WIDTH(L)
              ELSE
                A(L)=A(L)+WIDTH(L)
              END IF
            END IF
            MAGA2=MAGA2+A(L)**2
80    CONTINUE
          MAGA=SQRT(MAGA2)
          IF(MAGA.LE.SHELL)THEN
            DO 70 K=1,NATOMS
              IF(K.NE.I.AND.K.NE.J)THEN
                MAGB2=0.0
                MAGC2=0.0
                DO 60 L=1,3
                  B(L)=R(J,L)-R(K,L)
                  IF(ABS(B(L)).GT.(WIDTH(L)/2))THEN
                    IF(B(L).GT.0.0)THEN
                      B(L)=B(L)-WIDTH(L)
                    ELSE
                      B(L)=B(L)+WIDTH(L)
                    END IF
                  END IF
                  MAGB2=MAGB2+B(L)**2
                  C(L)=R(K,L)-R(I,L)
                  IF(ABS(C(L)).GT.(WIDTH(L)/2))THEN
                    IF(C(L).GT.0.0)THEN
                      C(L)=C(L)-WIDTH(L)
                    ELSE
                      C(L)=C(L)+WIDTH(L)
                    END IF
                  END IF
                  MAGC2=MAGC2+C(L)**2
60    CONTINUE
                MAGB=SQRT(MAGB2)
                MAGC=SQRT(MAGC2)
C
                WGHT1=(MAGA*MAGB)**4
                COSTH1=(A(1)*B(1)+A(2)*B(2)+A(3)*

```

```

*          B(3))/(MAGA*MAGB)
          IF(COSTH1.GT.1.0)THEN
            COSTH1=1.0
          ELSEIF(COSTH1.LT.-1.0)THEN
            COSTH1=-1.0
          END IF
          ANGLE1=ACOS(COSTH1)*180.0/PI
          U=NINT(ANGLE1)
          IF(MAGB.GT.SHELL)THEN
            BIN1(U)=BIN1(U)+1/WGHT1
          ELSE
            BIN2(U)=BIN2(U)+1/WGHT1
          END IF
C
          WGHT2=(MAGA*MAGC)**4
          COSTH2=(C(1)*A(1)+C(2)*A(2)+C(3)*
*          A(3))/(MAGC*MAGA)
          IF(COSTH2.GT.1.0)THEN
            COSTH2=1.0
          ELSEIF(COSTH2.LT.-1.0)THEN
            COSTH2=-1.0
          END IF
          ANGLE2=ACOS(COSTH2)*180.0/PI
          V=NINT(ANGLE2)
          BIN1(V)=BIN1(V)+1/WGHT2
          IF(MAGC.GT.SHELL)THEN
            BIN1(V)=BIN1(V)+1/WGHT1
          ELSE
            BIN2(V)=BIN2(V)+1/WGHT1
          END IF
          END IF
70      CONTINUE
          END IF
90      CONTINUE
100     CONTINUE
C
C      --OUTPUT--
      DO 200 I=0,180
        WRITE(20,*) I,BIN1(I),BIN2(I)
200     CONTINUE
C      --CLOSE FILES--

      ENDFILE(UNIT=20)
      CLOSE(UNIT=20)
C      --THE END--
      STOP
      END

```

```

PROGRAM ADF2
C
C   Program to find the ADF of a lattice generated by
C   the program Dynamo.
C
PARAMETER(MAXATMS=4000)
DIMENSION R(MAXATMS,3), BIN1(0:180), BIN2(0:180),
*A(3), B(3), C(3), WIDTH(3)
CHARACTER *15 OUTFILE,INPUT
CHARACTER *60 STRING1
REAL MAGA2, MAGB2, MAGC2, MAGA, MAGB, MAGC
INTEGER U, V
C
C   --OPEN FILES--
C
10  WRITE(6,*) 'ENTER OUTPUT FILENAME: '
    READ(5,20)OUTFILE
20  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=20, FILE=OUTFILE, STATUS='NEW', ERR=10)
30  WRITE(6,*) 'ENTER INPUT FILENAME: '
    READ(5,40)INPUT
40  FORMAT(A)
    WRITE(6,*)
    OPEN(UNIT=30, FILE=INPUT, STATUS='OLD', ERR=30)
C
C   --SET PARAMETERS--
C
WRITE(6,*) 'ENTER RADIAL SUPPRESSION CUTOFF THICKNESS'
READ(5,*) SHELL
C
C   --INPUT DATA FROM FILE--
READ(30,*)STRING1
READ(30,*)NATOMS,NTYPES,TRASH1
READ(30,*)WIDTH(1),WIDTH(2),WIDTH(3)
READ(30,*)TRASH2,TRASH3,TRASH4
DO 45 I=1,NTYPES
    READ(30,*)TRASH5,TRASH6
45  CONTINUE
DO 50 I=1,NATOMS
    READ(30,*)R(I,1),R(I,2),R(I,3)
    READ(30,*)TRASH7,TRASH8,TRASH9
    READ(30,*)TRASH10
50  CONTINUE
C
PI=ACOS(-1.0)
C
C
DO 100 I=1,NATOMS-1

```

```

DO 90 J=I+1,NATOMS
  MAGA2=0.0
  DO 80 L=1,3
    A(L)=R(J,L)-R(I,L)
    IF(ABS(A(L)).GT.(WIDTH(L)/2))THEN
      IF(A(L).GT.0.0)THEN
        A(L)=A(L)-WIDTH(L)
      ELSE
        A(L)=A(L)+WIDTH(L)
      END IF
    END IF
    MAGA2=MAGA2+A(L)**2
80  CONTINUE
  MAGA=SQRT(MAGA2)
  IF(MAGA.LE.SHELL)THEN
    DO 70 K=1,NATOMS
      IF(K.NE.I.AND.K.NE.J)THEN
        MAGB2=0.0
        MAGC2=0.0
        DO 60 L=1,3
          B(L)=R(J,L)-R(K,L)
          IF(ABS(B(L)).GT.(WIDTH(L)/2))THEN
            IF(B(L).GT.0.0)THEN
              B(L)=B(L)-WIDTH(L)
            ELSE
              B(L)=B(L)+WIDTH(L)
            END IF
          END IF
          MAGB2=MAGB2+B(L)**2
          C(L)=R(K,L)-R(I,L)
          IF(ABS(C(L)).GT.(WIDTH(L)/2))THEN
            IF(C(L).GT.0.0)THEN
              C(L)=C(L)-WIDTH(L)
            ELSE
              C(L)=C(L)+WIDTH(L)
            END IF
          END IF
          MAGC2=MAGC2+C(L)**2
60  CONTINUE
        MAGB=SQRT(MAGB2)
        MAGC=SQRT(MAGC2)
C
        WGHT1=(MAGA*MAGB)**4
        COSTH1=(A(1)*B(1)+A(2)*B(2)+A(3)*
          * B(3))/(MAGA*MAGB)
        IF(COSTH1.GT.1.0)THEN
          COSTH1=1.0
        ELSEIF(COSTH1.LT.-1.0)THEN
          COSTH1=-1.0

```



```

        END IF
        ANGLE1=ACOS(COSTH1)*180.0/PI
        U=NINT(ANGLE1)
        IF(MAGB.GT.SHELL)THEN
            BIN1(U)=BIN1(U)+1/WGHT1
        ELSE
            BIN2(U)=BIN2(U)+1/WGHT1
        END IF

C
        WGHT2=(MAGA*MAGC)**4
        COSTH2=(C(1)*A(1)+C(2)*A(2)+C(3)*
*      A(3))/(MAGC*MAGA)
        IF(COSTH2.GT.1.0)THEN
            COSTH2=1.0
        ELSEIF(COSTH2.LT.-1.0)THEN
            COSTH2=-1.0
        END IF
        ANGLE2=ACOS(COSTH2)*180.0/PI
        V=NINT(ANGLE2)
        BIN1(V)=BIN1(V)+1/WGHT2
        IF(MAGC.GT.SHELL)THEN
            BIN1(V)=BIN1(V)+1/WGHT1
        ELSE
            BIN2(V)=BIN2(V)+1/WGHT1
        END IF
        END IF
70      CONTINUE
        END IF
90      CONTINUE
100     CONTINUE
C
C      --OUTPUT--
      DO 200 I=0,180
          WRITE(20,*) I,BIN1(I),BIN2(I)
200     CONTINUE
C      --CLOSE FILES--

      ENDFILE(UNIT=20)
      CLOSE(UNIT=20)
C      --THE END--
      STOP
      END

```

10 Vita

Captain David Wesley was born on 15 August 1958 in Ventura, California. He graduated from Cloverdale High School in Cloverdale, California in 1976. After two years of college in California he enlisted in the U.S. Air Force and had several assignments before attending the Georgia Institute of Technology. He Graduated from Georgia Tech in September 1986 with a Bachelor of Nuclear Engineering. He then attended Officer Training School whereupon he received a regular commission in the USAF and an assignment to Kirtland AFB, New Mexico. He performed duties as an aircraft nuclear safety engineer for the Air Force Weapons Laboratory with primary responsibility for the F-16 until entering the School of Engineering, Air Force Institute of Technology, in August 1989.